

# FAQ

## FREQUENTLY ASKED QUESTIONS





FAQ

FAQ

*Doc. MS062101*  
*Ed. 1.2 - English - 11/08/2021*



## IMPORTANT

CMZ SISTEMI ELETTRONICI S.r.l. reserves the right to make changes to the products described in this document at any time without notice.

This document has been prepared by CMZ SISTEMI ELETTRONICI S.r.l. solely for use by its customers, guaranteeing that at the date of issue it is the most up-to-date documentation on the products.

Users use the document under their own responsibility and certain functions described in this document should be used with due caution to avoid danger for personnel and damage to the machines.

No other guarantee is therefore provided by CMZ SISTEMI ELETTRONICI S.r.l., in particular for any imperfections, incompleteness or operating difficulties.

This document contains confidential information that is proprietary to CMZ SISTEMI ELETTRONICI S.r.l.. Neither the document nor the information contained therein should be disclosed or reproduced in whole or in part, without express written consent of CMZ SISTEMI ELETTRONICI S.r.l..



## Warning about the contents

The contents of this document have not be intended as suggestion on how to face some particular situations that may happen during the use of a CMZ's product. It is not guaranteed that the reported suggestions will always be sufficient for the solution of the required topic.

Before to try to execute an action according to the here described instructions, be sure to respect all the safety precautions and the technical specifications of the product in question, reported in the related documents (that can be downloaded from the download area of the [www.cmz.it](http://www.cmz.it) website or be requested at the [support@cmz.it](mailto:support@cmz.it) address).

---

# Table of Contents

|  |           |
|--|-----------|
| <b>1. SD DRIVE .....</b>   | <b>1</b>  |
| Transformer for ISD drive .....  | 2         |
| Management of the hardware enable input on SVM .....                   | 3         |
| Management of the CUSTOM position capture .....                        | 4         |
| <b>2. BD DRIVE .....</b>   | <b>7</b>  |
| Electric gear management via bus .....                                 | 8         |
| Use of the IBD/NBD drives with TwinCAT .....                           | 9         |
| Management of velocity feed forward in mode 8 .....                    | 10        |
| Control supply voltage lacking during a saving of the parameters ..... | 11        |
| Management of the CUSTOM position capture .....                        | 12        |
| <b>3. SD SETUP .....</b>   | <b>15</b> |
| Warning message of Configuration file not updated .....                | 16        |
| Retentive variables .....  | 17        |
| Electric gear management via bus .....                                 | 18        |
| Firmware update .....  | 20        |
| Set the filter on an input .....                                       | 21        |
| <b>4. LBD .....</b>  | <b>23</b> |
| Management of the position capture .....                               | 24        |
| Management of the parameters files in the drive .....                  | 26        |
| Problem to save the parameters .....                                   | 27        |
| Management of the brake .....  | 28        |
| The following error does not decrease by adjusting the gains .....     | 30        |
| <b>5. EASY .....</b>   | <b>31</b> |
| DC bus in common .....   | 32        |
| <b>6. GEM DRIVE STUDIO .....</b>                                       | <b>33</b> |
| Configuration of the Siboni motors .....                               | 34        |

|  |           |
|--|-----------|
| Import of a motors list .....  | 35        |
| <b>7. FCT .....</b>  | <b>37</b> |
| Retentive memory in the FCT controllers .....                                    | 38        |
| Cloning of the application that is present in the controller .....               | 39        |
| Firmware update .....  | 40        |
| Modify the IP address .....  | 41        |
| Descriptor file of the FCT controller .....                                      | 43        |
| SD card compatibility .....  | 45        |
| Management of the retentive variables .....                                      | 47        |
| Writing in the SD card failure .....   | 49        |
| <b>8. CODESYS .....</b>  | <b>51</b> |
| Filter a value by using an array .....   | 52        |
| Management of a generic axis .....   | 53        |
| Sending of the master references to manage the electric gear via bus .....       | 54        |
| Notice on the firmware version difference between the used devices .....         | 56        |
| Notice on the library CMZ_HBus .....   | 57        |
| Management of a cam with the library CMZ_Cam .....                               | 58        |
| Management of the UPD communication with the library CMZ_WebServer .....         | 60        |
| Retentive variables and persistent variables .....                               | 61        |
| Firmware update of the drives .....  | 62        |
| Update of the device used in CODESYS .....                                       | 63        |
| Import of a new device .....   | 65        |
| Management of the CAN and ETC nodes startup .....                                | 67        |
| Link of a variable over a PDO .....  | 68        |
| Management of the strings in modbus .....  | 70        |
| Activation of the FTP server with library CMZ_FTPServer .....                    | 73        |
| User creation for the access through FTP server .....                            | 74        |
| Axis resolution .....  | 75        |
| Connection to the FCT without network scan .....                                 | 76        |
| Management of the modbus TCP (client FCT) with library CMZ_Modbus .....          | 78        |
| Management of the modbus TCP (server FCT) with library CMZ_Modbus .....          | 80        |
| Activation of the analog inputs of the WAGO module .....                         | 82        |
| Management of retentive modbus variables .....                                   | 83        |
| Comparison between two arrays .....  | 85        |
| Set, not automatically, the ID of an EtherCAT node .....                         | 86        |
| Activation of the web server and api management with library CMZ_WebServer ..... | 87        |
| H_Bus starting problems .....  | 90        |

|   |           |
|---|-----------|
| <b>9. HMI .....</b>                         | <b>91</b> |
| Retentive variables on HMI .....            | 92        |
| Communication between HMI and SDDrive ..... | 95        |
| Modify the keyboard dimension .....         | 97        |
| Trasfer of a project between two HMI .....  | 98        |





# Chapter 1

## SD DRIVE

| Code         | Description                                    |
|--------------|--|
| SDDRIVE_0001 | Transformer for ISD drive                      |
| SDDRIVE_0002 | Management of the hardware enable input on SVM |
| SDDRIVE_0003 | Management of the CUSTOM position capture      |

**Table 1.1. Arguments**

## Transformer for ISD drive

---

Transformer for ISD drive

### Question

Can a 35 VA transformer be suitable to supply an ISD?

### Answer

A 35 VA transformer is too little to supply an ISD.

To calculate the power that the transformer has to have is necessary to consider the formula:

$P_{TRASF} \sim PHVT / (0,7 * u)$  where:

- $P_{TRASF}$  is the transformer power.
- $u$  is the efficiency of the transformer (= 0.9).
- $PHVT$  is the total power that is absorbed by the ISD drives.

## Management of the hardware enable input on SVM

---

Management of the hardware enable input on SVM

### Question

Can the hardware enable input of the SVM drive be connected to a safe output?

### Answer

The hardware enable input can be connected to a safe output.

If this safe output has the diagnosis functionalities it is necessary to set, through SDSSetup, the filter on the enable input. The filter setting has to be made (through the diagnosis function) according to the electrical characteristics of the output to which it is connected, so that to remove the noises and guarantee the correct functioning of the input, without rebounds.

To set the filter on the input, refer to the question [Set the filter on an input](#)

## Management of the CUSTOM position capture

Management of the CUSTOM position capture

### Question

How shall the CUSTOM interface captures in the SDDrive drives be configured through the controller?

### Answer

In order to configure the captures, follow the hereafter described steps:

1. If it is necessary to use the PDOs, add between the PDOs of the SVM/ISD node the following cells:
  - 0x4001.01 : capture status if it is used the machine A;  
0x4011.01 : capture status if it is used the machine B;
  - 0x4004.01 : capture position if it is used the machine A;  
0 0x4014.01 : capture position if it is used the machine B;

|   |                                 |           |
|---|---------------------------------|-----------|
| <input checked="" type="checkbox"/> <b>16#1802: Param. 0x180200</b> | <b>16#384 (\$NODEID+16#380)</b> | <b>48</b> |
| Latch status A  | 16#4001:16#01                   | 16        |
| Latch value A   | 16#4004:16#01                   | 32        |

Figure 1.1. Example: PDO adding for the first capture machine

If it is not necessary to use the PDOs for the position capture management, skip this step.

2. 0x4000.02 : capture trigger signal configuration if it is used the machine A;  
0x4010.02 : capture trigger signal configuration if it is used the machine B;
3. 0x4003.01 : capture source configuration if it is used the machine A;  
0x4013.01 : capture source configuration if it is used the machine B;
4. 0x4000.03 : capture on rising/falling edge configuration if it is used the machine A;  
0x4000.03 : capture on rising/falling edge configuration if it is used the machine B;

5. To select the capture machine, enable the captures and read the captured value, use the *MC\_TouchProbe* function block. The capture machine must be selected through the *TriggerInput* input of the function block (TriggerInput = 0 : machine A; TriggerInput = 1 : machine B)..

For further information on the captures configuration, refer to the *SDDrive* manual.



## Chapter 2

# BD DRIVE

| Code         | Description  |
|--------------|--|
| BDDRIVE_0001 | Electric gear management via bus                                 |
| BDDRIVE_0002 | Use of the IBD/NBD drives with TwinCAT                           |
| BDDRIVE_0003 | Management of velocity feed forward in mode 8                    |
| BDDRIVE_0004 | Control supply voltage lacking during a saving of the parameters |
| BDDRIVE_0005 | Management of the CUSTOM position capture                        |

**Table 2.1. Arguments**

## Electric gear management via bus

---

Electric gear management via bus

### Question

How shall the electric gear be managed via bus in the IBD and NBD drives?

### Answer

In order to manage the electric gear via bus refer to:

- The *Electric gear management via bus* question for the electric gear management via programmability.
- The *Sending of the master references to manage the electric gear via bus* question for the CODESYS project configuration to send master position and velocity via bus.



---

## Use of the IBD/NBD drives with TwinCAT

---

Use of the IBD/NBD drives with TwinCAT

### Question

How shall the IBD and NBD drives be used with TwinCAT, the Beckhoff software?

### Answer

For the IBD and NBD drives management with TwinCAT refer to the *BDDrive* manual, on "*Example of drive connection by using TwinCAT*" paragraph.

## Management of velocity feed forward in mode 8

---

Management of velocity feed forward in mode 8

### Question

How shall the velocity feed forward in mode 8 be managed in the the drives commanded by the master?

### Answer

To manage the velocity feed forward in mode 8 it is necessary to use the transmission PDO to pass even the cell 0x60B1.00 (*VelocityOffset*).

This parameter can be used as velocity feed forward or velocity reference according to the interpolation mode (*SubMode*). By default the *SubMode* is -136, therefore the *VelocityFeedForward* calculation depends on the cell 0x60B1.00.

For further information refer to the manual *BDDrive*.

# Control supply voltage lacking during a saving of the parameters

Control supply voltage lacking during a saving of the parameters

## Question

What happens if, during the saving of the parameters in the permanent memory through the button *Save all parameters* from SDSetup, there is a lack of supply voltage in the control section?

## Answer

If during a saving of the parameters in the permanent memory there is a lack of the control supply voltage, it could happen the situation illustrated by the following image, losing the motor configuration:

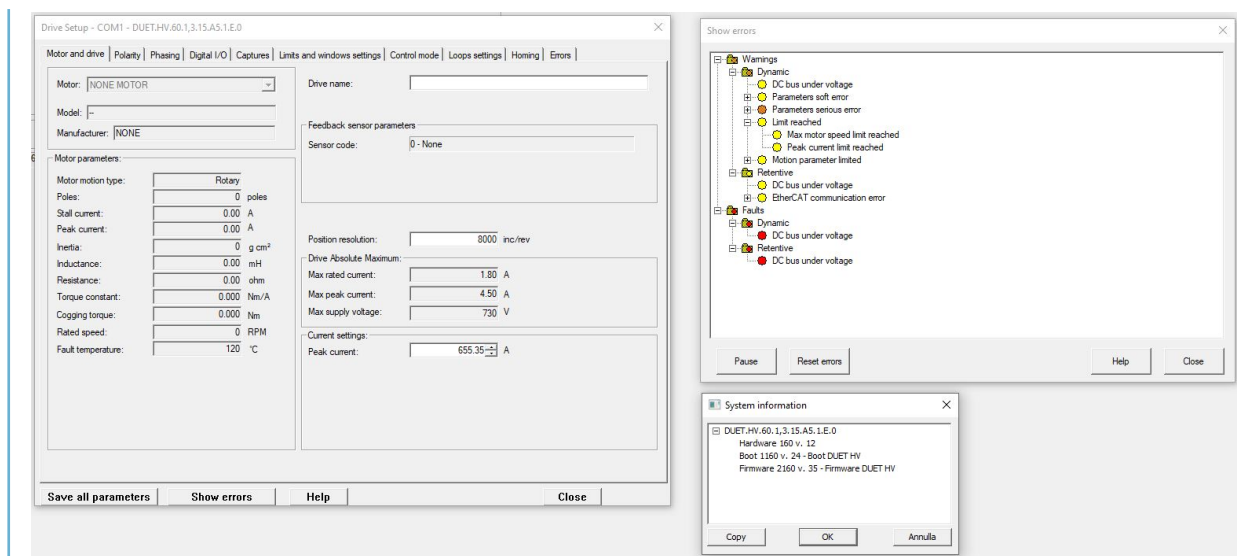


Figure 2.1. Errors during the parameters saving in the permanent memory

In these cases it is necessary to execute, through SDSetup, the restore of the factory default data by clicking the menu bar on *Drive-> Permanent memory...-> Restore default parameters*.

## Management of the CUSTOM position capture

Management of the CUSTOM position capture

### Question

How shall the CUSTOM interface captures in the BDDrive drives be configured through the controller?

### Answer

In order to configure the captures. follow the hereafter described steps:

1. If it is necessary to use the PDO, add the following cells between the TPDO of the node IBD/NBD:
  - 0x4001.01 : capture status if it is used the machine A;  
0x4011.01 : capture status if it is used the machine B;
  - 0x4004.01 : capture position if it is used the machine A;  
0 0x4014.01 : capture position if it is used the machine B;

|                                     |  |                                 |           |
|-------------------------------------|--|---------------------------------|-----------|
| <input checked="" type="checkbox"/> | <b>16#1802: PdoTx3_CommunicationParameters</b> | <b>16#381 (\$NODEID+16#380)</b> | <b>48</b> |
|                                     | Latch status A                                 | 16#4001:16#01                   | 16        |
|                                     | Latch value A                                  | 16#4004:16#01                   | 32        |

Figure 2.2. Example: PDO adding for the first capture machine

If it is not necessary to use the PDOs for the position capture management, skip this step.

2. 0x4000.02 : capture trigger signal configuration if it is used the machine A;  
0x4010.02 : capture trigger signal configuration if it is used the machine B;
3. 0x4003.01 : capture source configuration if it is used the machine A;  
0x4013.01 : capture source configuration if it is used the machine B;
4. 0x4000.03 : capture on rising/falling edge configuration if it is used the machine A;

5. To select the capture machine, enable the captures and read the captured value, use the *MC\_TouchProbe* function block. The capture machine must be selected through the *TriggerInput* input of the function block (TriggerInput = 0 : machine A; TriggerInput = 1 : machine B)..

For further information on the captures configuration, refer to the *BDDrive* manual.



## Chapter 3

# SD SETUP

| Code         | Description                                       |
|--------------|---|
| SDSETUP_0001 | Warning message of Configuration file not updated |
| SDSETUP_0002 | Retentive variables                               |
| SDSETUP_0003 | Electric gear management via bus                  |
| SDSETUP_0004 | Firmware update                                   |
| SDSETUP_0005 | Set the filter on an input                        |

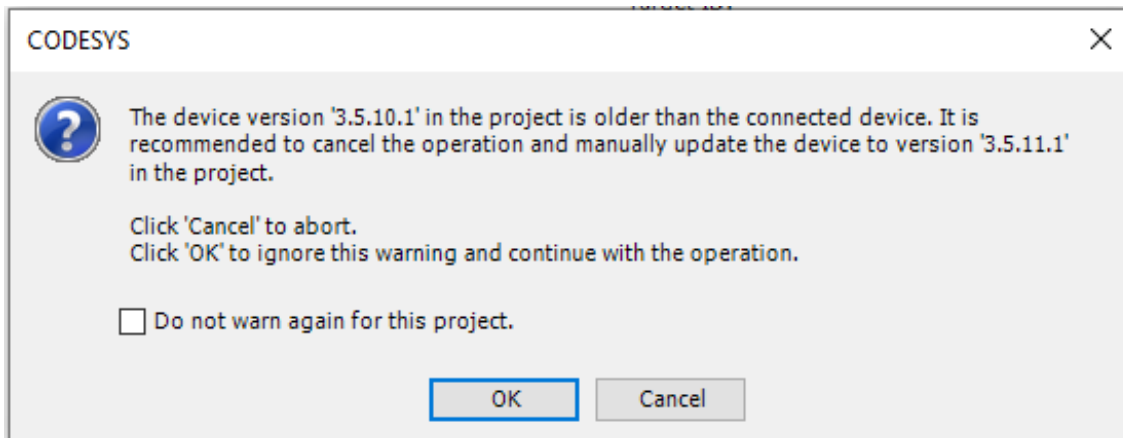
**Table 3.1. Arguments**

## Warning message of Configuration file not updated

Warning message of *Configuration file not updated*

### Question

What shall be done when this message appears at the SDSetup startup?



### Answer

This message indicates that the version of the xml file in the drive is not present in the files that are saved in the PC.

By answering YES to this message, the connection will be made with the xml file that is stored in the PC that has the version that is the most near to the one in the drive.

To not see this message again and to obtain the required xml file, it is necessary to update SDSetup.



## Retentive variables

---

Retentive variables

### Question

Why does the retentive variables not work in the programmability?

### Answer

To use the retentive variables it is necessary to activate them by writing 1 in the modbus cell 588.

This cell must be written in the reset program.

Furthermore, in the drives of the SD series it is necessary to have a supply of at least 40 V in order to use the retentive variables.

If the retentive variables are activated and the drive is not correctly supplied, the drive will not turn-on.

## Electric gear management via bus

---

Electric gear management via bus

### Question

How shall the electric gear be managed via bus in the IBD and NBD drives, through the internal programmability?

### Answer

In order to manage the electric gear refer to:

```
VAR_GLOBAL
  AxisSlave           : AXIS_REF;
  EncoderMaster      : ENC_REF;
  Fb_Start           : Mc_Start;
  1 Fb_Stop          : MC_Stop;
  Fb_Power           : Mc_Power;
  Fb_Gear            : MC_Gear;

  xStartGear         : BOOL := FALSE;
  xStopGear          : BOOL := FALSE;
  StepCycle          : SINT := 0;
  Vel_Stop           : DINT := 160000;
END_VAR
```

```

PROGRAM main

AxisSlave.Num := MC_REF_AXIS_MAIN;
EncoderMaster.Num:= IO_REF_ENC_AUXILIARY; ❷

CASE StepCycle OF
0:
;
1:
SYS_WriteObject(6524, 2);
SYS_WriteObject(6537, 1);
SYS_WriteObject(6539, 2);
SYS_WriteObject(6554, 0); ❸

StepCycle := StepCycle + 1;

2:
IF xStartGear THEN
SYS_WriteObject(6534, 1); ❹
Fb_Gear.InpStart := TRUE;
Fb_Gear.StartGear := TRUE;
StepCycle := StepCycle + 1;
END_IF;

3:
IF xStopGear THEN
Fb_Stop.Execute := TRUE;
Fb_Gear.StartGear := FALSE;
SYS_WriteObject(6534, 0); ❺
StepCycle := StepCycle + 1;
END_IF;

4:
IF Fb_Stop.Done THEN
Fb_Stop.Execute := FALSE;
StepCycle := 2;
END_IF
END_CASE

Fb_Start(Execute := TRUE);

Fb_Stop(Axis := AxisSlave,
Deceleration := Vel_Stop);

Fb_Power(Axis := AxisSlave);

Fb_Gear(Master := EncoderMaster,
Slave := AxisSlave,
Mode := 2,
RatioInNumerator := 1000, ❻
RatioInDenominator := 1000,
RatioEndNumerator := 1000,
RatioEndDenominator := 1000,
MasterSpace := 1);

END_PROGRAM
    
```

- ❶ Variables declaration and instantiation of the function block.
- ❷ Select the type of the encoder that is used as master, that is the auxiliary encoder (IO\_REF\_ENC\_AUXILIARY).
- ❸ Writing of the following cells to set some encoder settings:
  - 6524: Select the auxiliary encoder type (2=Fieldbus Auxiliary Encoder).
  - 6537: Select the functioning mode of the fieldbus auxiliary encoder (1=Pos+Vel with active extrapolator).
  - 6539: Selection of the fieldbus auxiliary encoder extrapolation period, expressed in sync periods number.
  - 6554: Enable=1, disable=0 of the output parameters "BusAuxEncoderOutPosition" and "BusAuxEncoderOutVelocity" of the fieldbus auxiliary encoder.
- ❹ Writing of the cell 6534 to switch the fieldbus auxiliary encoder to run mode.
- ❺ Writing of the cell 6534 to switch the fieldbus auxiliary encoder to stop mode.
- ❻ Use the function block *MC\_Gear* to manage the electric gear, through the master, the slave, the gear mode, the initial and final following ratio and the master position within which the slave has to reach the final following ratio.

For further information about the cells refer to the manual *BDDrive*.

For the configuration of the CODESYS project that allows to send the master position and velocity via bus, refer to the question [Sending of the master references to manage the electric gear via bus](#)

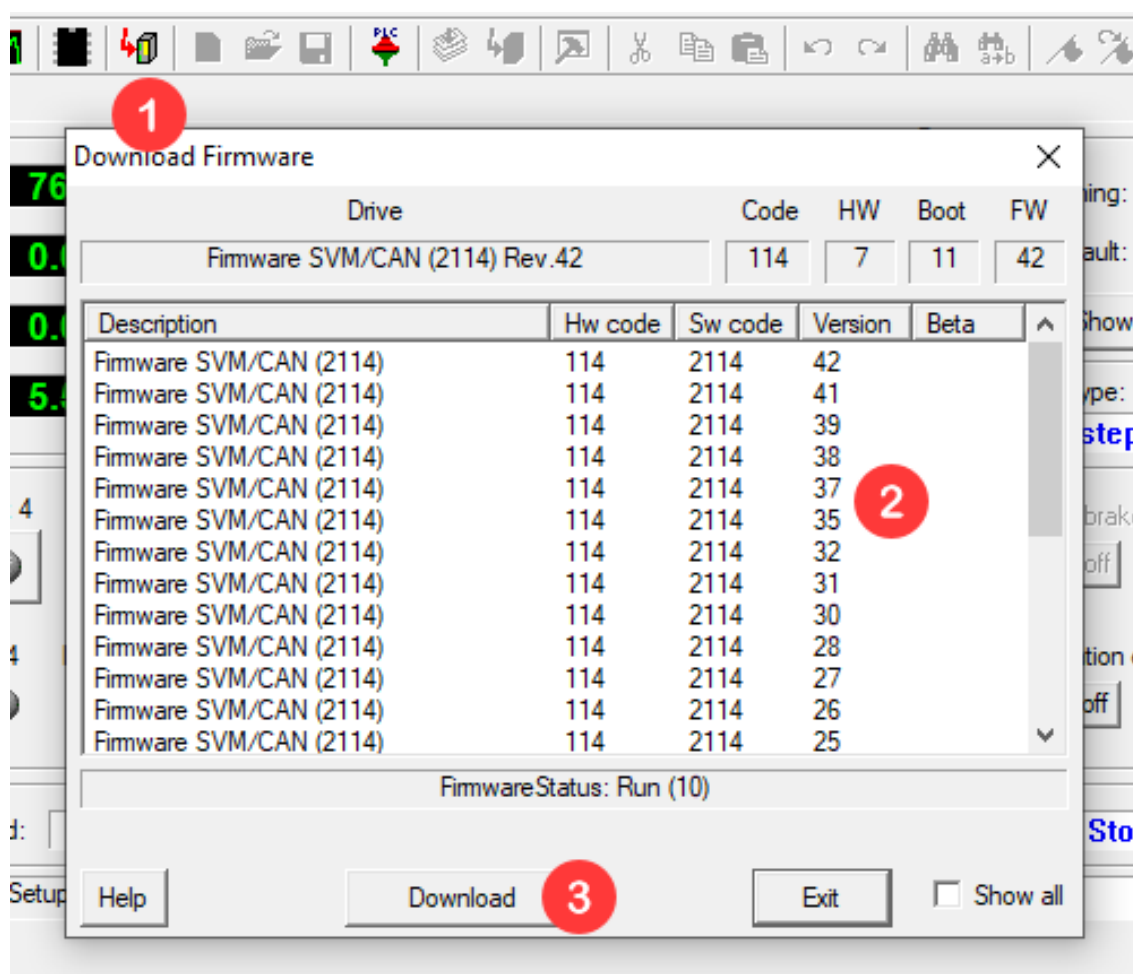
## Firmware update

Firmware update

### Question

How shall the SDDrive and BDDrive drives firmware be updated?

### Answer



- 1 Click on the icon *Download firmware*.
- 2 Select the firmware version to be downloaded in the drive.
- 3 Click on *Download* to start the firmware download procedure.

## Set the filter on an input

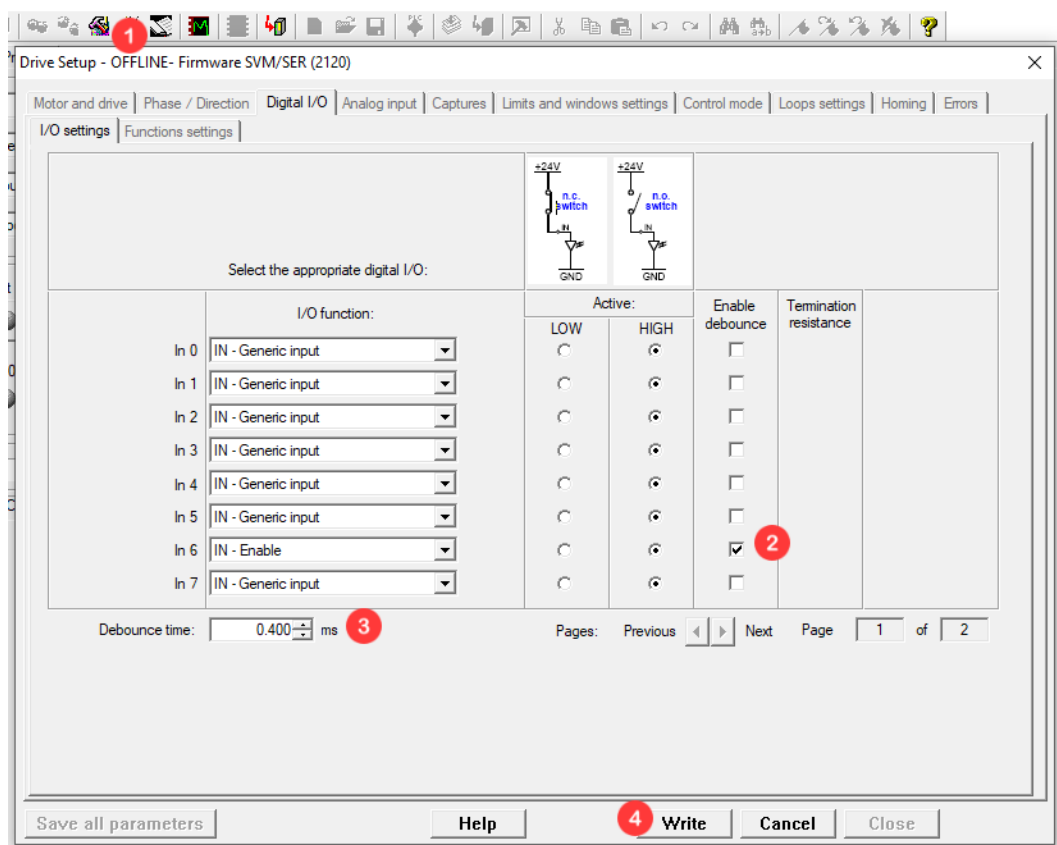
Set the filter on an input

### Question

How shall the filter be set on an input?

### Answer

To set the filter on an input follow the hereafter described steps:



- 1 From the window *Drive Setup* enter the tab *Digital I/O*.
- 2 Enable the filter through the check on *Enable debounce*.
- 3 Set the validation time of the input through *Debounce time*.

The inputs are scanned every  $200 \mu\text{s}$  and the time that is set on *Debounce time* is internally converted in sample number, therefore the expressed value has to be divisible by  $200 \mu\text{s}$ .

The minimum value that can be set is  $400 \mu\text{s}$ , while the maximum value is 3 s.



| Code     | Description  |
|----------|--|
| LBD_0001 | Management of the position capture                           |
| LBD_0002 | Management of the parameters files in the drive              |
| LBD_0003 | Problem to save the parameters                               |
| LBD_0004 | Management of the brake                                      |
| LBD_0005 | The following error does not decrease by adjusting the gains |

**Table 4.1. Arguments**

## Management of the position capture

Management of the position capture

### Question

How shall the captures be configured in the LBD drive through controller?

### Answer

In order to configure the captures, follow the hereafter described steps:

1. If it is necessary to use the PDOs, add the following cells among the TPDOs of the LBD node:
  - 0x3370.00 : capture status;
  - 0x337X.06 where the X stands for the capture machine to be used (3371.06 : machine 1; 3372.06 machine 2, etc): captured position.

In the following image the TPDO that refers to the 4th capture machine has been added:

| POSITION VALUE   | ADDRESS                  | SIZE |
|--|--------------------------|------|
| <input checked="" type="checkbox"/> 16#1803: TPDO4 Parameter | 16#481 (\$NODEID+16#480) | 48   |
| Captures status  | 16#3370:16#00            | 16   |
| Capture 4 Position   | 16#3374:16#06            | 32   |

If it is not necessary to use the PDOs for the position capture management, skip this step.

2. Between the initialization SDOs (Node LBD -> Tab SDOs) add the selection of the capture trigger, by using the cell 0x337X.03, where X stands for the capture machine to be used.

In the following image the SDO used to configure the input 4 as capture input in the 4th machine has been added:

|    |               |                 |      |    |                          |                          |   |  |
|----|---------------|-----------------|------|----|--------------------------|--------------------------|---|--|
| 52 | 16#3374:16#03 | Capture 4 Input | 16#3 | 16 | <input type="checkbox"/> | <input type="checkbox"/> | 0 |  |
|----|---------------|-----------------|------|----|--------------------------|--------------------------|---|--|

In4

3. The SDO that configure the capture source is added by default, among the initialization SDOs, when the LBD node is inserted. This SDO selects the capture source (axis position) for the first capture machine. Therefore, if another machine is used, it is necessary to change the index of the SDO cell that has been inserted by default (0x337X.02 where X stands for the used machine).



In the following image the SDO to configure the trigger source of the 4th capture machine has been modified:

|    |               |                  |             |    |                          |                          |   |  |
|----|---------------|------------------|-------------|----|--------------------------|--------------------------|---|--|
| 44 | 16#3374:16#02 | Capture 1 Source | 16#60640000 | 32 | <input type="checkbox"/> | <input type="checkbox"/> | 0 |  |
|----|---------------|------------------|-------------|----|--------------------------|--------------------------|---|--|

4. To select the capture machine, enable the captures and read the captured value, use the *MC\_TouchProbe* function block. The capture machine must be selected through the *TriggerInput* input of the function block.

For further informations on the capture configuration, refer to the *LBD\_User\_Manual* manual, chapter 3.2.5.5 and paragraph *Capture Parameters*.

## Management of the parameters files in the drive

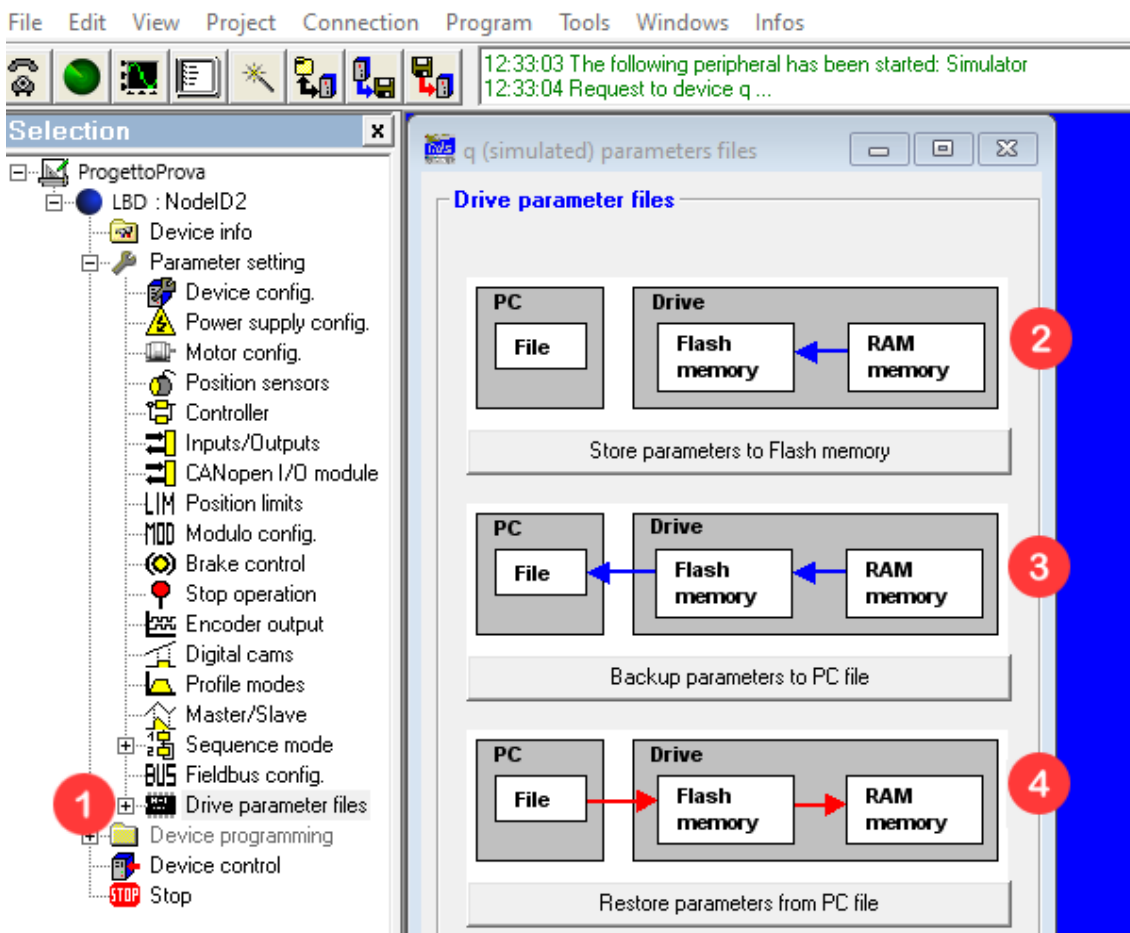
Management of the parameters files in the drive

### Question

How shall a parameters file be imported in the drive?

### Answer

To save the parameters in the drive there are two procedures:



- 1 From the navigation tree of the project push on *Drive parameter files*.
- 2 *Store parameter to Flash memory* : it allows to save, in a permanent way in the drive, the parameters set from GemDriveStudio.
- 3 *Backup parameters to PC file* : it allows to save, in a permanent way in the drive, the parameters set from GemDriveStudio and to save the parameter file in the project folder.
- 4 *Restore parameters from PC file* : it allows to save, in a permanent way in the drive, the parameters that are present in a parameter file that already exists in the PC.

# Problem to save the parameters

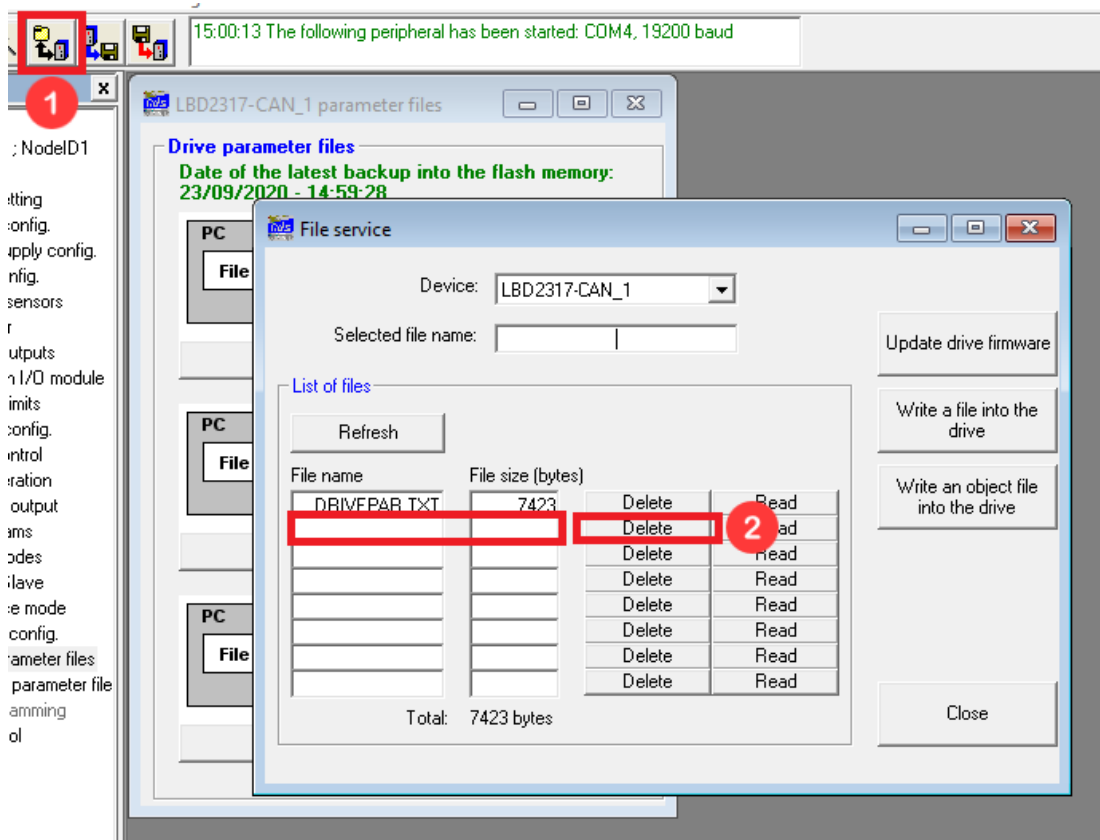
Problem to save the parameters

## Question

Why do some parameters cannot be saved in the permanent memory of the drive?

## Answer

A reason why the some parameters cannot be saved in the permanent memory is that in the drive an user parameters file is already present that overwrites some parameters. To delete this file it is necessary to:



- 1 Click on the *File service* button, as in the figure.
- 2 Check that below the *DRIVEPAR.TXT* there is another file. If it is present, delete it through the *Delete* button.

**ATTENTION : do not delete the file *DRIVEPAR.TXT*, but the successive one!**

## Management of the brake

Management of the brake

### Question

How shall the brake be managed in the LBD drives?

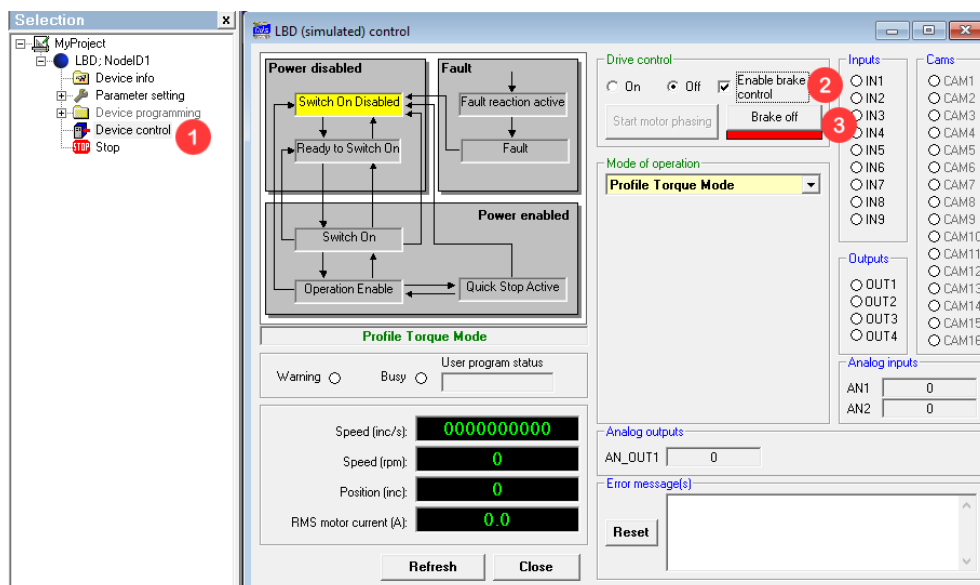
### Answer

The brake management is, by default, automatic and the brake automatically intervenes when the axis is disabled.

To manually manage the brake through SDO it is necessary:

- Write 1 in the cell 60FE.02.
- Activate or deactivate the brake through the bit 0 of the cell 60FE.01.
- To return to the automatic management of the brake it is necessary to write 0 in the cell 60FE.02.

To manually manage the brake thorough GemDriveStudio it is necessary to:



- 1 From the project tree double click on *Device control*.
- 2 To enable the manual management of the brake check the option *Enable brake control*.
- 3 Activate or deactivate the brake through the button *Brake off/Brake on*.

**Note**

In the LBD230 V drives it is necessary to configure a digital output for the brake management, while in the LBD400 V it is not necessary because there is a reserved output.

## The following error does not decrease by adjusting the gains

The following error does not decrease by adjusting the gains

### Question

Why, while correctly tuning the motor and modifying the gains, is it not possible to reduce the following error? see [Figure 4.1](#)

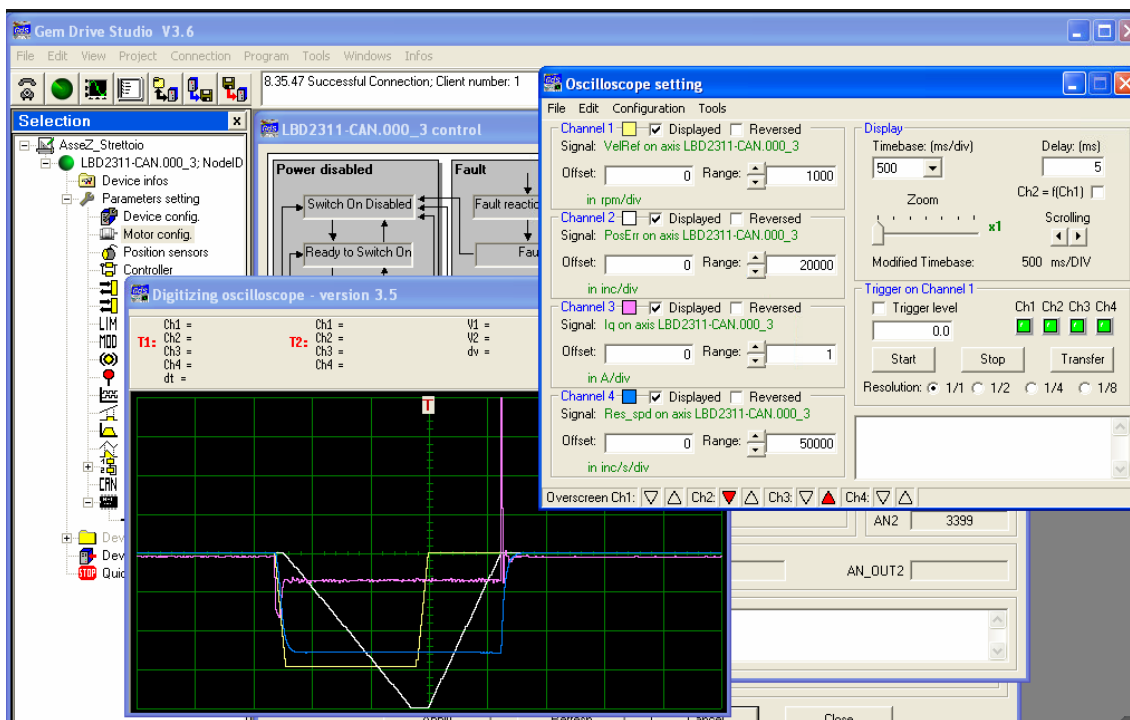


Figure 4.1. Following error chart during the movement

### Answer

As reported in the figure, with the increasing of the velocity, and in relation to an high following error, the applied current remains below 1 A and this behaviour doesn't change even if the gains are increased, as if there is a saturation that limits the current to 1 A. While at low velocities both the motor and the drive correctly react, because by increasing the current the following error becomes zero. This behaviour was caused by the connection of a 400 V motor to a drive of 230 V. So, if the motor cannot be correctly tuned, it is advisable to check that the association between the LBD and the motor is correct (LBD40 with motor 400 V and LBD23 with motor 230 V).

| Code      | Description      |
|-----------|------------------|
| EASY_0001 | DC bus in common |

**Table 5.1. Arguments**

## DC bus in common

DC bus in common

### Question

Is it possible to link the EASY drive DC bus and so to use a single braking resistance?

### Answer

Yes, it is possible. For the connections please refer to the table below:

#### 3.6.1 - XtrapulsEasy™-ak-230/17: X4

Manufacturer: Weidmüller

Type: BLZ 5.08 / 8

Reference: 152706

Tightening torque: 0.4 to 0.5Nm

| PIN | SIGNAL | I/O | FUNCTION                                    | DESCRIPTION  |
|-----|--------|-----|---|--|
| 1   | U      | O   | Motor phase U                               | Shielded motor cable:<br>- PE connection on the bottom plate,<br>- 360° shield connection. |
| 2   | V      | O   | Motor phase V                               |  |
| 3   | W      | O   | Motor phase W                               |  |
| 4   | DC-    | I/O | DC bus negative voltage output              | For the DC bus paralleling in multi-axis applications                                      |
| 5   | DC+    | I/O | DC bus positive voltage output              |  |
| 6   | DR     | O   | Braking transistor output                   | Minimum braking resistor value = 50Ω<br>Connect the braking resistor between pins 5 and 6. |
| 7   | L1     | I   | 230V <sub>AC</sub> single-phase mains input | 230V <sub>AC</sub> single-phase +10% / -15%<br>Fully integrated EMC mains filter.          |
| 8   | L2     | I   | supply                                      |  |

Collegare la resistenza di frenatura tra DC+ e DR

ATTENZIONE! Il valore di resistenza non deve essere inferiore a 50 ohm



# GEM DRIVE STUDIO

| Code                     | Description                        |
|--------------------------|------------------------------------|
| GEMDRIVES-<br>TUDIO_0001 | Configuration of the Siboni motors |
| GEMDRIVES-<br>TUDIO_0002 | Import of a motors list            |

**Table 6.1. Arguments**

## Configuration of the Siboni motors

---

Configuration of the Siboni motors

### Question

How shall the Siboni motors be configured in Gem Drive Studio?

### Answer

For the configuration of the Siboni motors it is necessary to refer to the manual *SiboniMotors-GDS\_ApplicationNote*.

## Import of a motors list

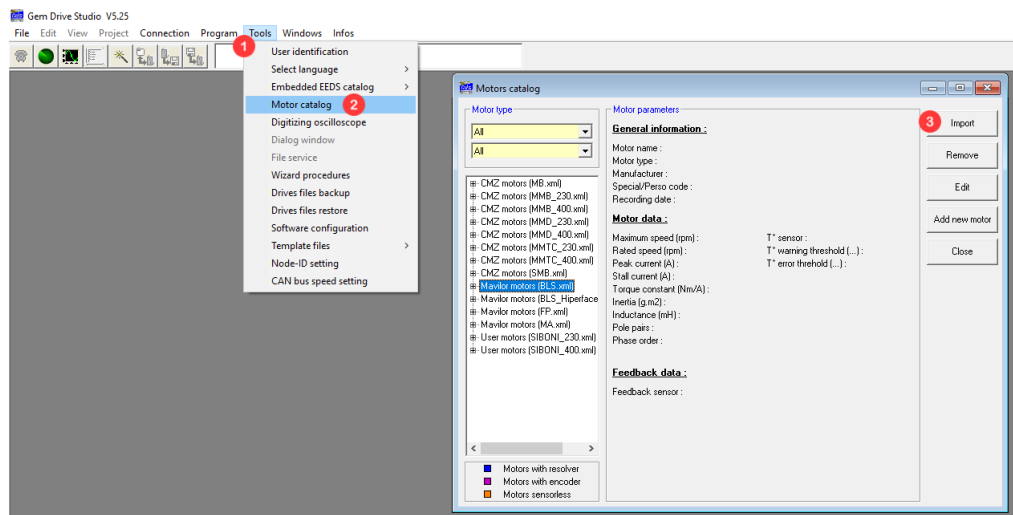
Import of a motors list

### Question

How shall a motors list be imported?

### Answer

To import a motors list in Gem Drive Studio follow the hereafter described procedure:



- 1 From the manu bar of Gem Drive Studio click on *Tools*.
- 2 Click on *Motor catalog*.
- 3 Click on *Import* and select the motors list to be imported.



| <b>Code</b> | <b>Description</b>   |
|-------------|--|
| FCT_0001    | Retentive memory in the FCT controllers                      |
| FCT_0002    | Cloning of the application that is present in the controller |
| FCT_0003    | Firmware update  |
| FCT_0004    | Modify the IP address  |
| FCT_0005    | Descriptor file of the FCT controller                        |
| FCT_0006    | SD card compatibility  |
| FCT_0007    | Management of the retentive variables                        |
| FCT_0008    | Writing in the SD card failure                               |

**Table 7.1. Arguments**

## Retentive memory in the FCT controllers

---

Retentive memory in the FCT controllers

### Question

How much memory is reserved for the retentive/persistent variables in the FCT640, FCT300 and FCT200 with CODESYS?

### Answer

The retentive memory reserved is 30 kB for the FCT640 and the FCT200, while it is 120 kB for the FCT300.

## Cloning of the application that is present in the controller

---

Cloning of the application that is present in the controller

### Question

How shall the cloning of the application that is present in the controller be done through FCT-Tool?

### Answer

To clone the application with FCTTool follow the hereafter described steps:

1. From the *Target locator* tab, select the controller that contains the application to be cloned.
2. Click on *Terminal*.
3. Enter the *CODESYS SoftPLC* page by pushing F1.
4. Enter the *Runtime Menu Info* page by pushing F2.
5. Enter the *Software info page* by pushing F1.
6. Push F10 to clone the application.

The procedure might not be immediate, but it may take time.

## Firmware update

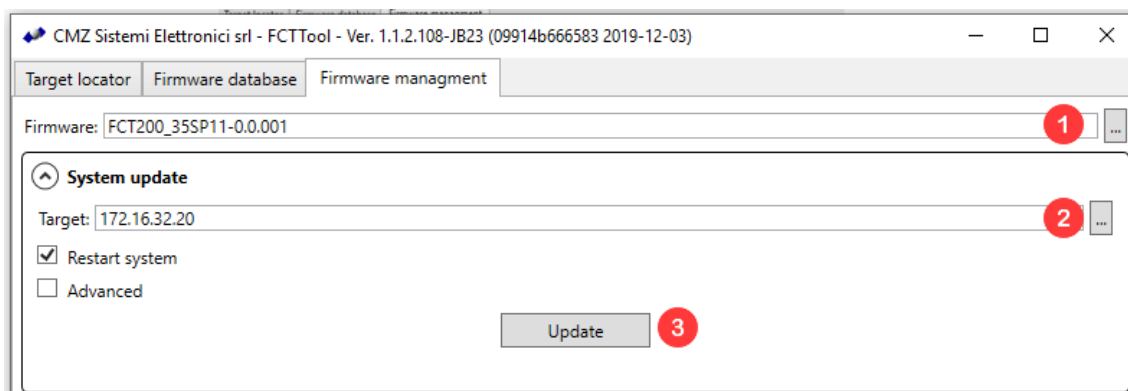
Firmware update

### Question

How shall the FCT controllers firmware be updated?

### Answer

To update the controllers firmware through FCTTool enter the *Firmware management* tab and follow the hereafter described steps:



- 1 Select the firmware to be downloaded from the database.

The service pack to be downloaded must be first imported in the database through the *Firmware database* tab by clicking on *Import...* and selecting the firmware to be imported.

- 2 Select the device to download the firmware in.
- 3 Click on *Update* to update the firmware.



# Modify the IP address

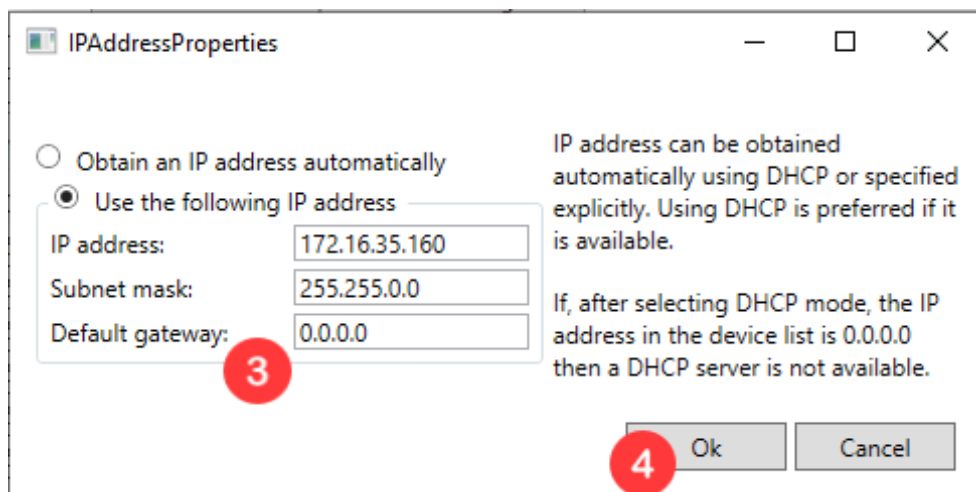
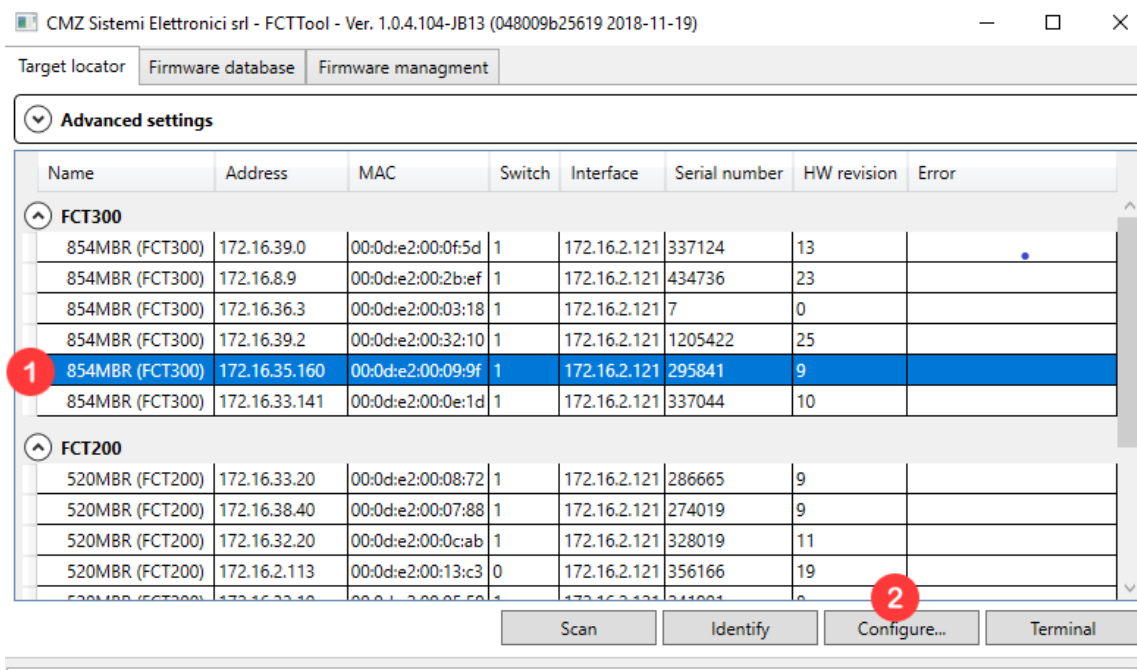
Modify the IP address

## Question

How shall the IP address be modified in the controllers through FCTTool?

## Answer

To modify the IP address from FCTTool follow the hereafter described steps:



- 1 From the *Target locator* tab select the device on which the IP address has to be changed.
- 2 Click on *Configure...*
- 3 Write the new IP address and the subnet mask.
- 4 Click *Ok* and if the procedure has been successfully executed, turn off and on again the system.

## Descriptor file of the FCT controller

Descriptor file of the FCT controller

### Question

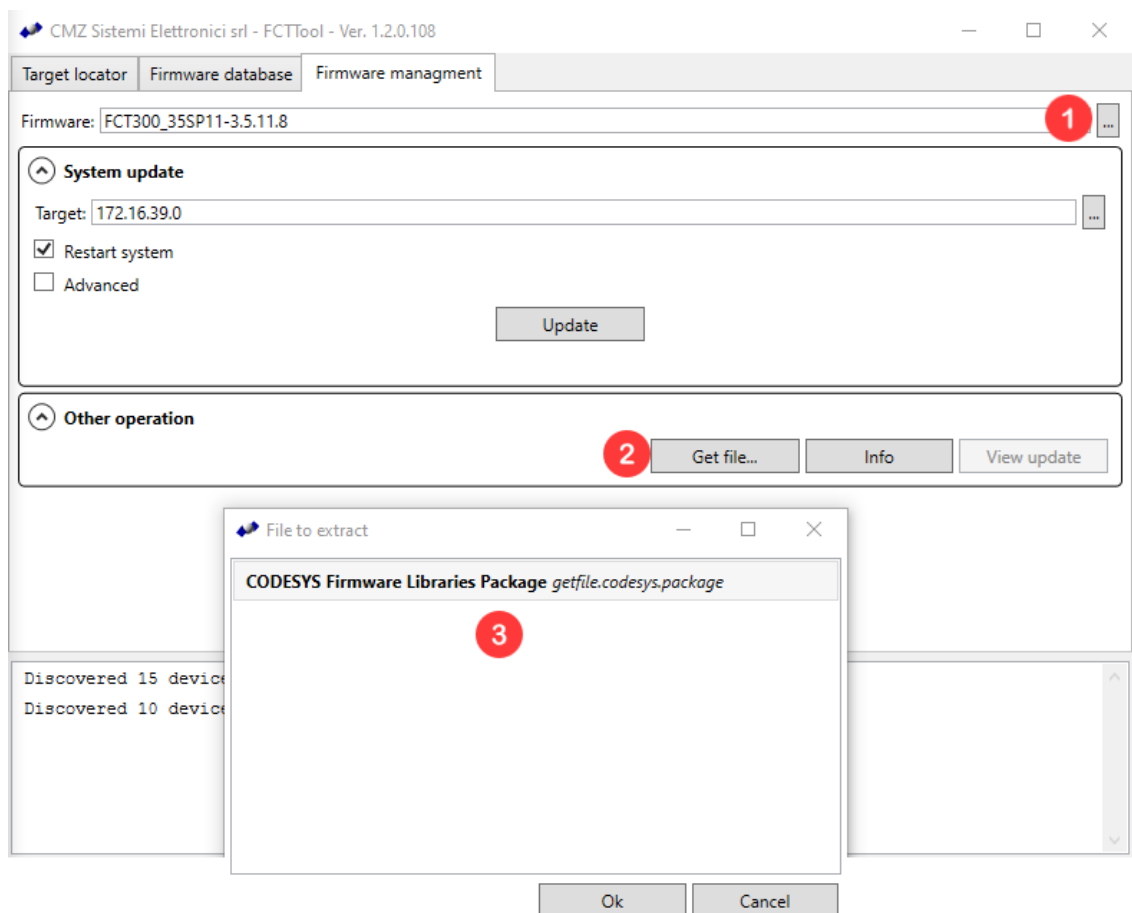
Where the controllers descriptor file can be found?

### Answer

The descriptor file can be found:

- In the service pack, that is the folder that contains the FCT firmware, the firmware libraries and the descriptor files.
- Can be exteact through the FCTToll if the service pack is imported in the database.

The procedure for the extraction is the following:



- 1 From the *Firmware management* tab select the service packs from which extract the file.
- 2 Click on *Get file...*
- 3 Select the pack that contains the descriptor files of the controller and the firmware libraries and save it in the desired path. Then double click on the pack to install it.

The extracted descriptor files must be imported in CODESYS in order to use this controller as device. See the question *Import of a new device*.

## SD card compatibility

SD card compatibility

### Question

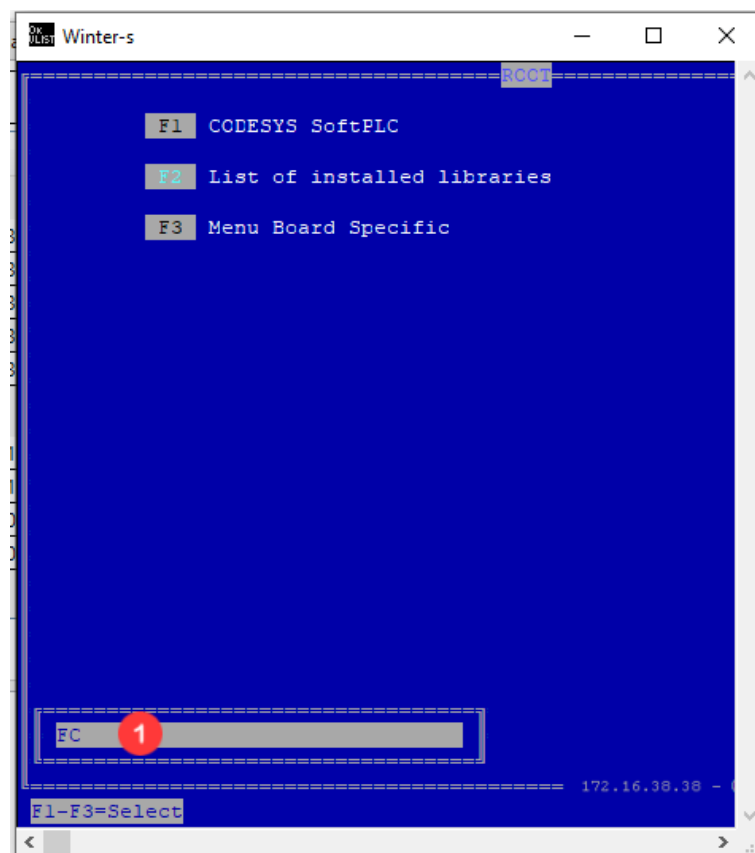
Which are the SD cards compatible with the FCT controllers?

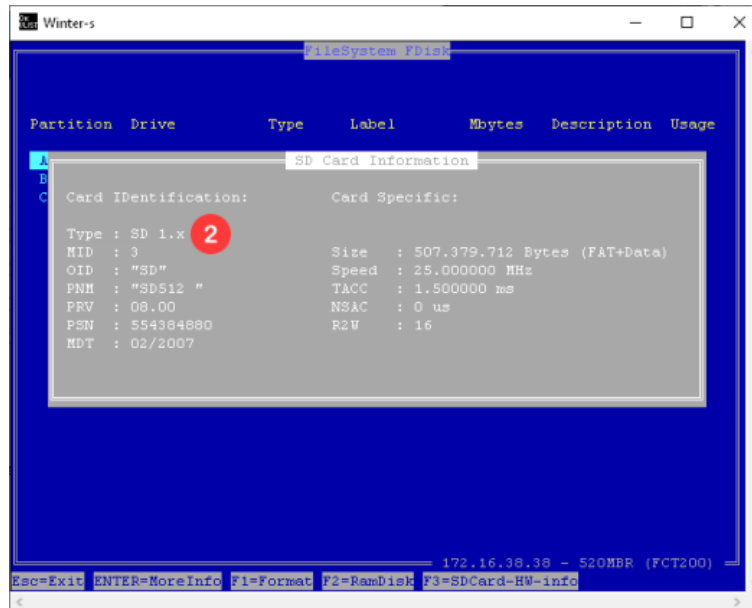
### Answer

In the FCT640 system only the SD Card that respect the specifications 2.0 and successive can work, while in the FCT200 and FCT300 can be used even the SD card of 1.0 type.

CMZ recommends to use the S-250 series of Swissbit SD cards, after having tested their compatibility with the controller and by recognizing a performance that is suitable with industry about temperature range and data retention reliability (SLC technology). The available sizes are 512 MB, 1 GB and 2 GB and it is possible to purchase the 1 GB version directly from CMZ.

To verify the type of an SD card, through the FCT200 or FCT300 systems, follow these steps:





- 1 From FCTool open the terminal and write *FC* to enter the *Filesystem Commander*.
- 2 From the page that opens with the command *FC*, push *ALT + F3* and then *F3 "SD-Card-HW-info"*.

From the just opened page, on the characteristic *Type* there is the inserted SD card type.

# Management of the retentive variables

Management of the retentive variables

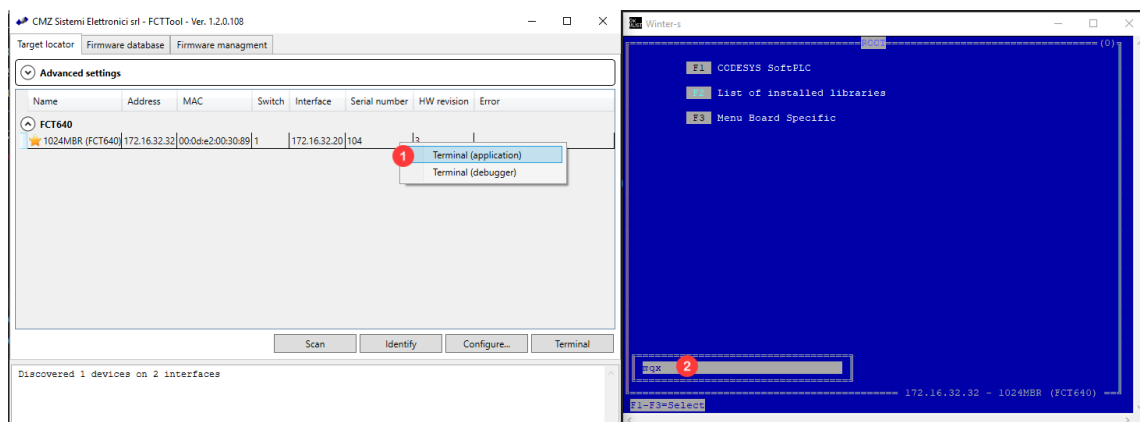
## Question

How shall:

- the retentive variables be saved in a file?
- the retentive variables be copied in the system from a file?
- the retentive variables be reset in a system?

## Answer

To manage the retentive variables it is necessary to use the FCTTool and follow the hereafter described steps:



```

===== MQX 5.0.0 Operating System =====
  F1 Task Summary
  3 F2 Kernel Memory Usage
  F3 Kernel I/O Devices
  F4 Kernel Time
  F5 Kernel Info
  F6 Kernel Test
  F7 Memory Dump
===== 172.16.32.32 - 1024MBR (FCT640) =====
Esc=Exit F1-F7=Select

```

```

===== Kernel Memory Usage =====
System available: 0x00000000 0x1FFFFFFF 0x20000000 512,0 MB
  F4
=====
Firmware Summary
-----
      Start      End      Size  Bytes  Perc
Code : 0x00400000 0x0079E93C 0x0039E93C 3,6 MB 52%
RoData: 0x0079E940 0x009B31B8 0x00214878 2,0 MB 30%
Data : 0x009B31C0 0x009B9CB4 0x00006AF4 26,7 KB 0%
Bss : 0x009B9D80 0x00AFEC40 0x00144EC0 1,2 MB 18%
Kernel available: 0x00AFEC40 0x1FFFFFFF 0x1F5013BF 501,0 MB
=====
Retain memory image
-----
      Start      End      Size  Bytes  Perc
Available: 0x00380000 0x00387FFF 0x00008000 32,0 KB
Kernel usage: 0x00380000 0x00380400 0x00000400 1,0 KB 3%
=====
Dynamic memory allocation
-----
      Start      End      Size  Bytes  Perc
Sys Mem Pool: 0x00AFF0C0 0x1FFFFFF40 0x1F500F00 501,0 MB
Highest Used Mem: 0x0111F97F 0x006208BF 6,1 MB 1%
Actual Used Mem: 0x01058C00 0x00550C00 5,3 MB 1%
Mem Pool Error:
Blocks:
  Num  Min Size  Max Size  Size  Bytes  Perc
Alloc: 1114 0x00000080 0x00318FC0 0x00611C00 6,0 MB 1%
Free : 22 0x00000080 0x1EEE05C0 0x1EEEF280 494,9 MB 98%
===== 172.16.32.32 - 1024MBR (FCT640) =====
Esc=Back

```

- 1 open the terminal.
- 2 Write *MQX*.
- 3 Push *F2* to enter the page *Kernel Memory Usage*.
- 4 From the page *Kernel Memory Usage* push:
  - Shift + *F8*: saves the retentive variables in a file called *UsrRet.bin* in the partition B of the system.
  - Shift + *F9*: copies the retentive variables in the system from a file called *UsrRet.bin* present in the partition B of the system.
  - Shift + *F10*: resets the retentive area.



## Writing in the SD card failure

Writing in the SD card failure

### Question

Why does the writing on SD card fail?

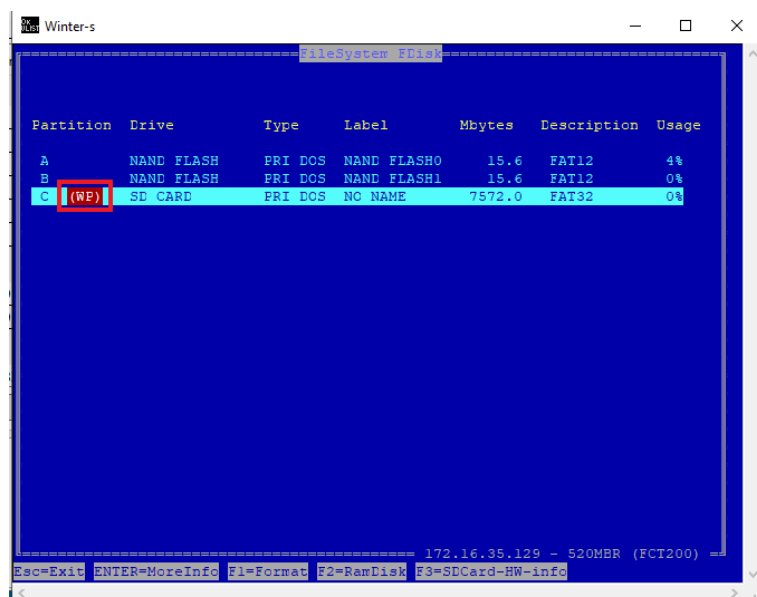
### Answer

The writing on SD card may fail due to:

- in the SD card the selector for the protected writing is active;
- in the SD card the selector for the protected writing is not active, but from FCTTool it results as it is active. In this case there is a hardware problem in the controller that concerns the SD card reader.

To verify from FCTTool that the SD card reader in the controller has correctly read the status of the selector that has been set on the SD card:

1. write *FC* to access to *Filesystem commander*;
2. push *ALT + F4* to access to the various partitions of the controller;
3. select the partition *C* that regards the SD card;
4. if the selector of the protected writing is active in the SD card, between the SD card details, it must appear the following writing:



Viceversa, if the protected writing selector is not active in the SD card, the *WP* initials must not appear between the SD card details.

## Chapter 8

# CODESYS

| Code         | Description  |
|--------------|--|
| CODESYS_0001 | Filter a value by using an array   |
| CODESYS_0002 | Management of a generic axis   |
| CODESYS_0003 | Sending of the master references to manage the electric gear via bus       |
| CODESYS_0004 | Notice on the firmware version difference between the used devices         |
| CODESYS_0005 | Notice on the library CMZ_HBus   |
| CODESYS_0006 | Management of a cam with the library CMZ_Cam                               |
| CODESYS_0007 | Management of the UPD communication with the library CMZ_WebServer         |
| CODESYS_0008 | Retentive variables and persistent variables                               |
| CODESYS_0009 | Firmware update of the drives  |
| CODESYS_0010 | Update of the device used in CODESYS                                       |
| CODESYS_0011 | Import of a new device   |
| CODESYS_0012 | Gestione dello startup e reset dei nodi CAN e ETC                          |
| CODESYS_0013 | Link of a variable over a PDO  |
| CODESYS_0014 | Management of the strings in modbus  |
| CODESYS_0015 | Activation of the FTP server with library CMZ_FTPServer                    |
| CODESYS_0016 | User creation for the access through FTP server                            |
| CODESYS_0017 | Axis resolution  |
| CODESYS_0018 | Connection to the FCT without network scan                                 |
| CODESYS_0019 | Management of the modbus TCP (client FCT) with library CMZ_Modbus          |
| CODESYS_0020 | Management of the modbus TCP (server FCT) with library CMZ_Modbus          |
| CODESYS_0021 | Activation of the analog inputs of the WAGO module                         |
| CODESYS_0022 | Management of retentive modbus variables                                   |
| CODESYS_0023 | Comparison between two arrays  |
| CODESYS_0024 | Set, not automatically, the ID of an EtherCAT node                         |
| CODESYS_0025 | Activation of the web server and api management with library CMZ_WebServer |
| CODESYS_0026 | H_Bus starting problems  |

**Table 8.1. Arguments**

## Filter a value by using an array

Filter a value by using an array

### Question

How is it possible to filter a value by using an array?

### Answer

To filter a value by using an array refer to the following image *see Figure 8.1*.

```
//Function block to filter the data
FUNCTION_BLOCK DataFilter
VAR_INPUT
  Enable      : BOOL;           //Enable the execution of function block
  ActualData  : LREAL;         //Actual data
END_VAR
VAR_OUTPUT
  FilteredData : LREAL;        //Filtered data
  xError       : LREAL;        //The error between actual data and filtered data
END_VAR
VAR
  Step      : INT;
  Full      : BOOL;
  ArrayData : ARRAY [0..MAXINDEX] OF LREAL;
  Indice    : INT := 0;
  Somma    : LREAL;
END_VAR
VAR CONSTANT
  MAXINDEX : INT := 9;         //Index of the array
END_VAR

IF Enable THEN
  IF Indice > MAXINDEX THEN           //if the index is greater than the value defined in MAXINDEX, the array is full.
    Full := TRUE;
    Indice := 0;
  END_IF
  IF Indice <= MAXINDEX AND Indice >= 0 THEN
    Somma := Somma-ArrayData[Indice]; //subtract from the sum the data of the array in the current index, in this way i will always have the sum of the last 10 data
    ArrayData[Indice] := ActualData; //i fill the array
    Somma := Somma+ArrayData[Indice]; //addition to the sum the last data acquired
    Indice := Indice +1; //increase the index
    IF NOT Full THEN
      FilteredData := Somma/Indice; //if the array hasn't yet been completely filled (10 elements) i divide the sum by the value of index
    ELSE
      FilteredData := Somma/(MAXINDEX-1); //if the array is complete i divide by ten
    END_IF
    xError := FilteredData-ActualData; //error between the filtered data and the current value
  END_IF
ELSE
  FilteredData := 0;
  Somma := 0;
  Indice := 0;
  xError := 0;
  Full := FALSE;
END_IF
```

Figure 8.1. Example of a function block to filter a value with an array

# Management of a generic axis

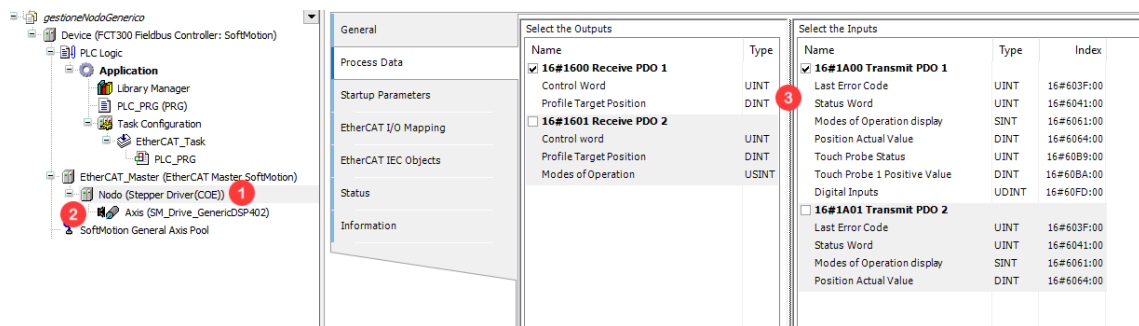
Management of a generic axis

## Question

How shall a generic axis be managed?

## Answer

To manage an axis with a generic driver it is necessary:



- 1 Add the node that has to be managed
- 2 Add an axis by right clicking on the added node and select *Add SoftMotion CiA402 Axis*.
- 3 Check the PDOs configuration from the node and verify that at least the ones that allow the motion are present (controlword, statusword and target position).

# Sending of the master references to manage the electric gear via bus

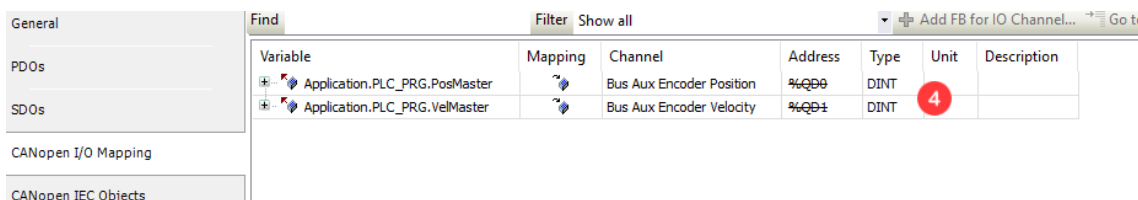
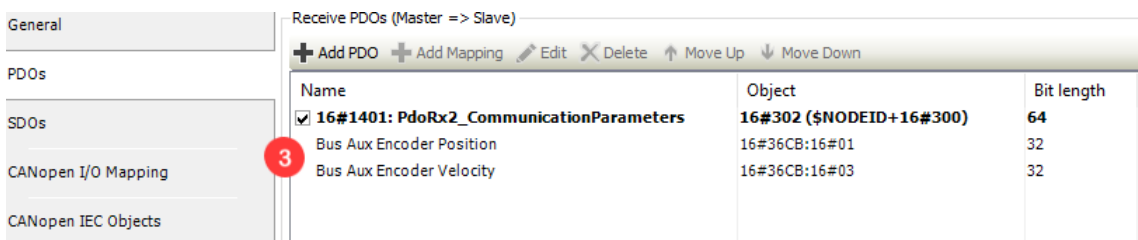
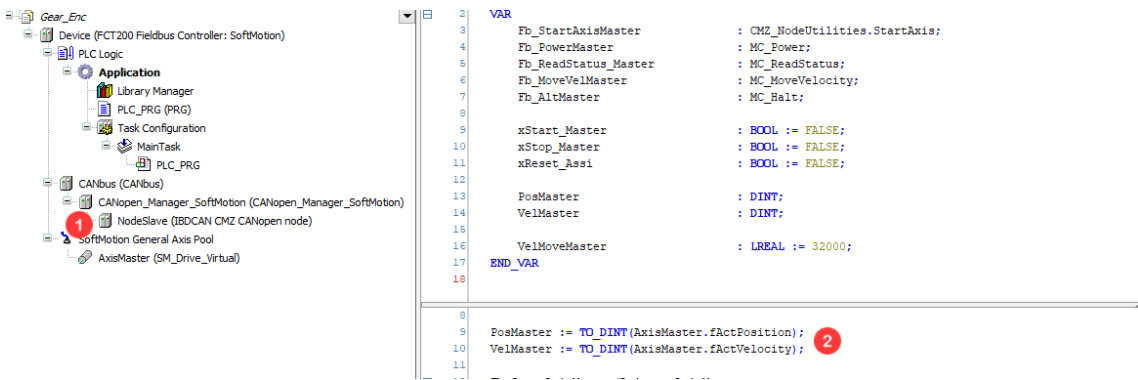
Sending of the master references to manage the electric gear via bus

## Question

What shall be configured, in the CODESYS project, to send the master position and velocity in order to manage the electric gear via bus?

## Answer

To send through bus the master position and velocity to manage the electric gear it is necessary to:



- 1 add an IBD or NBD node.
- 2 Get the master position and velocity.
- 3 Add between the receive PDOs the cells 36CB.01 (master position) and 36CB.03 (master velocity).

- 4 Map position and velocity in the added PDOs.

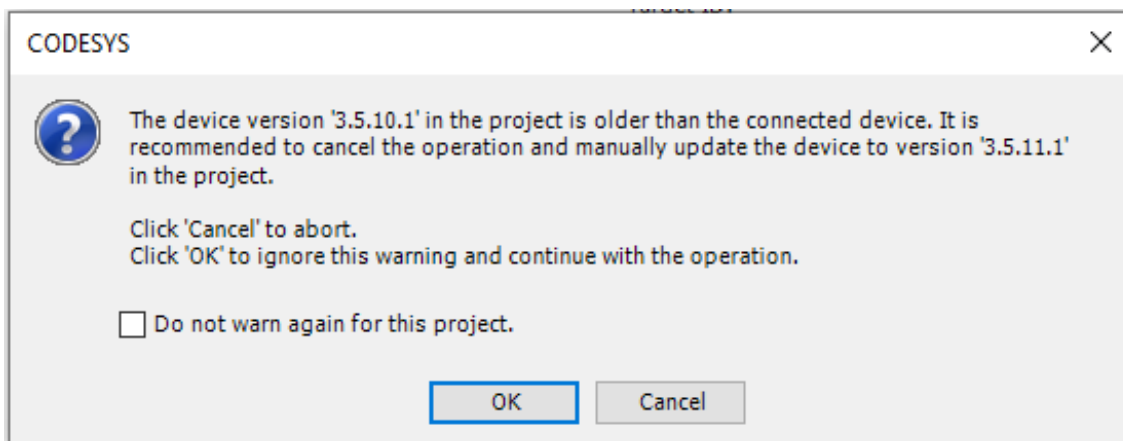
## Notice on the firmware version difference between the used devices

---

Notice on the firmware version difference between the used devices

### Question

Why does, during the project download, this message appears?



### Answer

This message appears when in a CODESYS project it is used a device version different from the firmware version inside the FCT.

It is therefore necessary to make the two versions match, by updating the version of the device that is used in CODESYS *see [Management of the CAN and ETC nodes startup](#)*.



## Notice on the library CMZ\_HBus

---

Notice on the library *CMZ\_HBus*

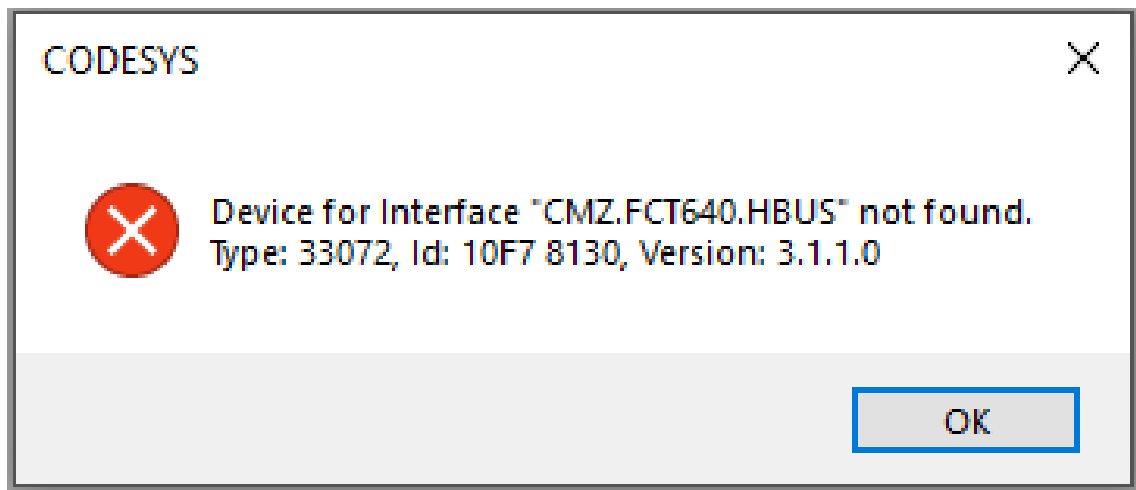
### Question

Why does not the HBUS appear in a project created with the FCT640?

### Answer

The HBUS does not appear when a CODESYS project is created with the FCT640, without having installed before the *CMZ\_HBUS* library contained in the service pack.

But the following error appears:



# Management of a cam with the library CMZ\_Cam

Management of a cam with the library *CMZ\_Cam*

## Question

What is a basic cam and how is it managed by using the library *CMZ\_Cam*?

## Answer

The cams are used to coordinate two axes, master and slave, where the slave movement depends on the master position and the path to be followed is defined by a specific table declared as array of type *CMZ\_Cam.Table* see *Figure 8.2*.

```

VAR
    CameTable      : ARRAY[0..9] OF CMZ_Cam.Table;
END_VAR

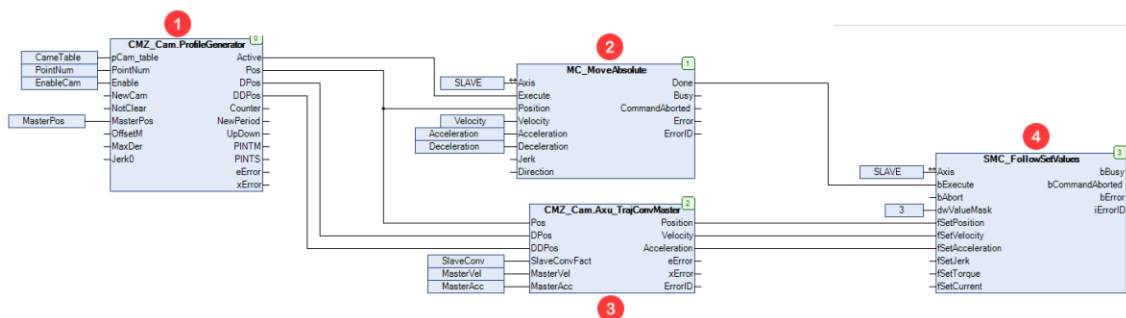
CameTable[0].Master := 0.0;
CameTable[0].Slave := 0.0;
CameTable[0].Der := 1.0;
CameTable[0].TypRamp :=1;

CameTable[1].Master := 4.0;
CameTable[1].Slave := 4.0;
CameTable[1].Der := 1.0;
CameTable[1].TypRamp :=1;

CameTable[2].Master := 7.0;
CameTable[2].Slave := 10.0;
CameTable[2].Der := 1.0;
CameTable[2].TypRamp :=1;
    
```

Figure 8.2. Example of a table for a 3 points cam

To manage a basic cam it is necessary to use:



- 1 The function block *ProfileGenerator*, present in the library *CMZ\_Cam* , that allows to generate the profile of a cam by receiving the previously described cam table in the input.
- 2 The CODESYS standard function block *MoveAbsolute*, that allows to move the axis in an absolute position passed as input by the user. This function block is used with the cams to correctly move the slave according to the cam table, before to set to the slave the values related to the cam profile.
- 3 The *Axu\_TrajConvMaster* function, present in the library *CMZ\_Cam* , that allows to convert Pos, DPos, DDPos in position, velocity and acceleration for the slave axis.
- 4 The standard function block of CODESYS *SMC\_FollowSetValues* that allows the axis to follow the profile given as input. It is used with the cams to set the position, velocity and acceleration in the slave axis.

# Management of the UPD communication with the library CMZ\_WebServer

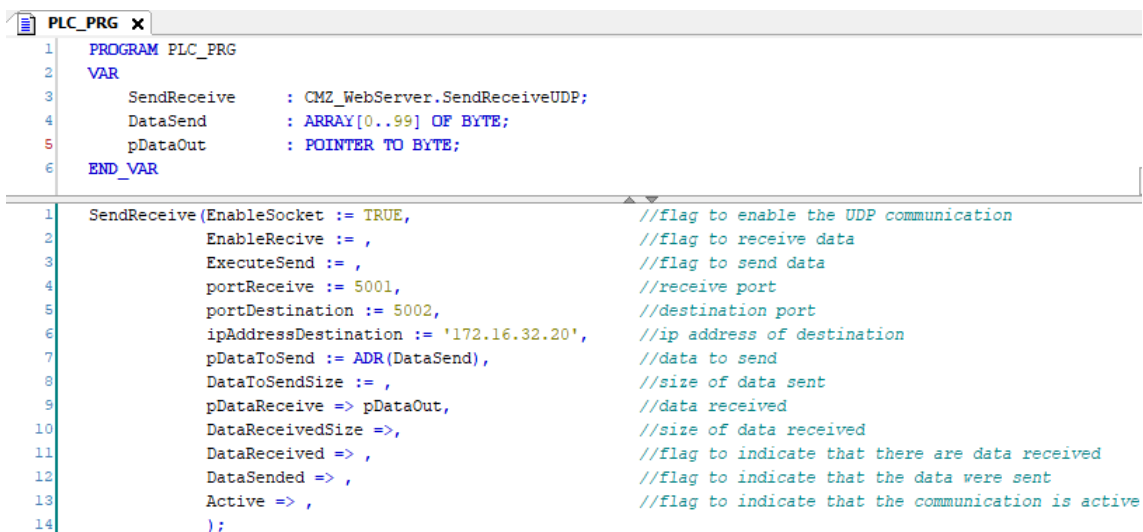
Management of the UPD communication with the library CMZ\_WebServer

## Question

How shall an UPD communication Client/Server in CODESYS be managed by using the library CMZ?

## Answer

To manage an UDP communication client/server in CODESYS it is necessary to import in the project the library *CMZ\_WebServer* and to use the function block *SendReceiveUDP* that is inside it see [Figure 8.3](#)



```
1 PROGRAM PLC_PRG
2 VAR
3     SendReceive      : CMZ_WebServer.SendReceiveUDP;
4     DataSend         : ARRAY[0..99] OF BYTE;
5     pDataOut         : POINTER TO BYTE;
6 END_VAR

1 SendReceive(EnableSocket := TRUE,           //flag to enable the UDP communication
2             EnableReceive := ,             //flag to receive data
3             ExecuteSend := ,              //flag to send data
4             portReceive := 5001,          //receive port
5             portDestination := 5002,      //destination port
6             ipAddressDestination := '172.16.32.20', //ip address of destination
7             pDataToSend := ADR(DataSend), //data to send
8             DataToSendSize := ,          //size of data sent
9             pDataReceive => pDataOut,    //data received
10            DataReceivedSize =>,         //size of data received
11            DataReceived => ,            //flag to indicate that there are data received
12            DataSended => ,              //flag to indicate that the data were sent
13            Active => ,                  //flag to indicate that the communication is active
14            );
```

Figure 8.3. Example of the function block SendReceiveUDP use

---

## Retentive variables and persistent variables

---

Retentive variables and persistent variables

### Question

What is the difference between retentive variables and persistent variables?

### Answer

- *Retentive variables*: these are variables that maintain their value even after an unexpected or desired turn off of the controller or when a reset warm command is executed.  
The retentive variables are re-initialized when an origin reset command is executed, and differently from the persistent variables, when a cold reset command or during an application download is executed.
- *Persistent variables*: these are retentive variables that can be re-initialized only through an origin reset command or through the download of a new application.

## Firmware update of the drives

Firmware update of the drives

### Question

How shall the drives firmware be updated through CODESYS?

### Answer

The drives firmware can be updated through the CODESYS program, by using the function block *CANopen\_DriveUpdater* for the drives CANopen or the function block *EtherCAT\_DriveUpdater* for the drives EtherCAT, that are present in the library *CMZ\_NodeUtilities*.

```

UpdaterFile x
1 PROGRAM UpdaterFile
2 VAR
3     DownloadFirmware : CMZ_NodeUtilities.CANopen_DriveUpdater;
4     ParamFile        : STRING;
5 END_VAR

1 DownloadFirmware(Axis := Axis,
2                 Network := 1,
3                 Node := LBD,
4                 Execute := DownloadGo,           //Flag to allow to start with the download of the chosen files (firmware, IEC, parameter file)
5                                                     //This input is the output "CallUpdater" of StartAxis function block
6                 ParamFile := ParamFile,         //Firmware name to download (in .bin format)
7                 ParamUpdate := TRUE,           //Flag to indicate that must firmware download
8                 Done => DownloadDone          //Flag to indicate that the firmware download is finished
9                                                     //This output must be connect to the input "UpdaterDone" of the StartAxis function block
10            );

```

Figure 8.4. Example of use of the function block that manages the firmware download

Using the same function block it is possible to download in the drive even the parameter file, passing the the parameters file name to the input *ParamFile* and setting the input *ParamUpdate*. Furthermore it is possible to download the IEC program (internal programmability) passing the parameters file name to the input *IECFile* and setting the input *IECUpdate*.



#### Note

Instance and use this function block in a proper task, different from the current one, that contains the motion programs.

## Update of the device used in CODESYS

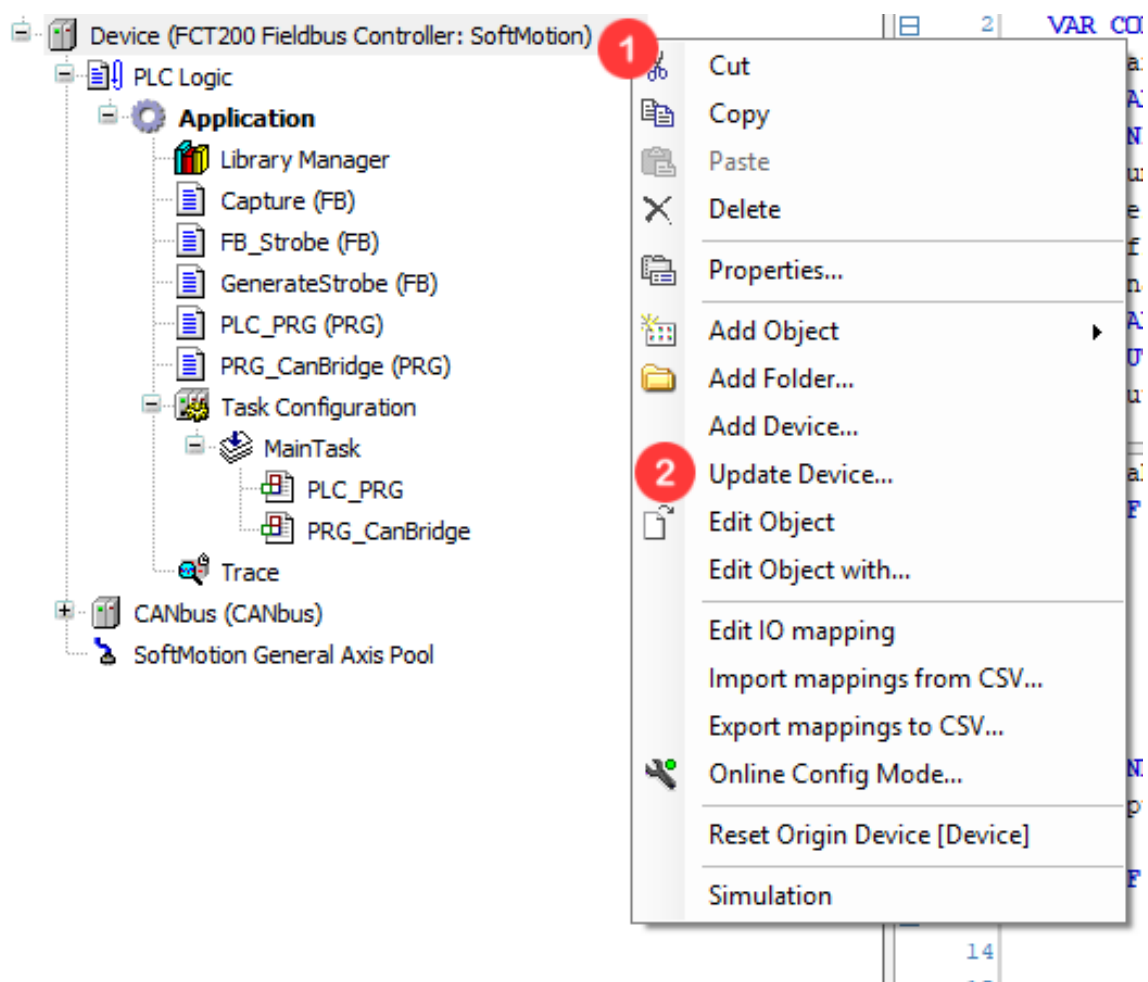
Update of the device used in CODESYS

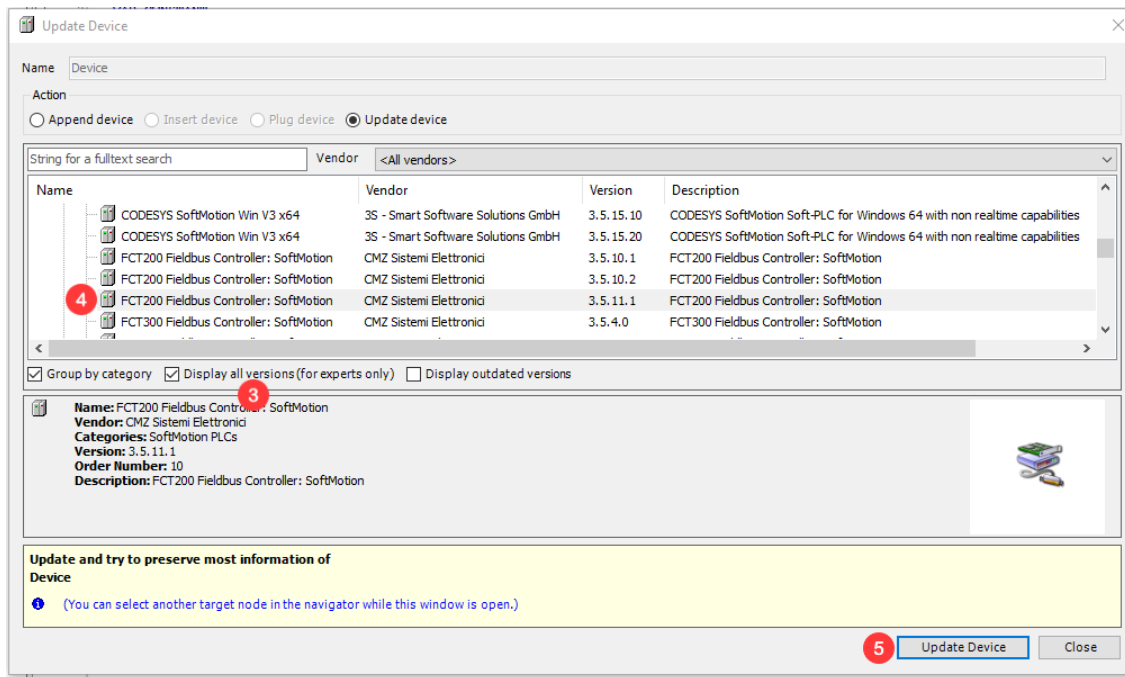
### Question

How shall the device used in CODESYS be updated?

### Answer

To update the version of the device used in CODESYS follow the hereafter described steps:





- 1 Right click on *Device*.
- 2 Left click on *Update Device...*
- 3 From the window *Update Device* check the *Display all versions* box to see all the versions imported in the device.
- 4 Select the device version that has to be used.
- 5 Left click on *Update Device*.

N.B: If the device version that has to be used is not present in the list, it is necessary, before to do the previously described steps, to import in CODESYS the device description file with the version that has to be used.

For the explanation about how to import in CODESYS the description file, refer to the question [Link of a variable over a PDO](#)



## Import of a new device

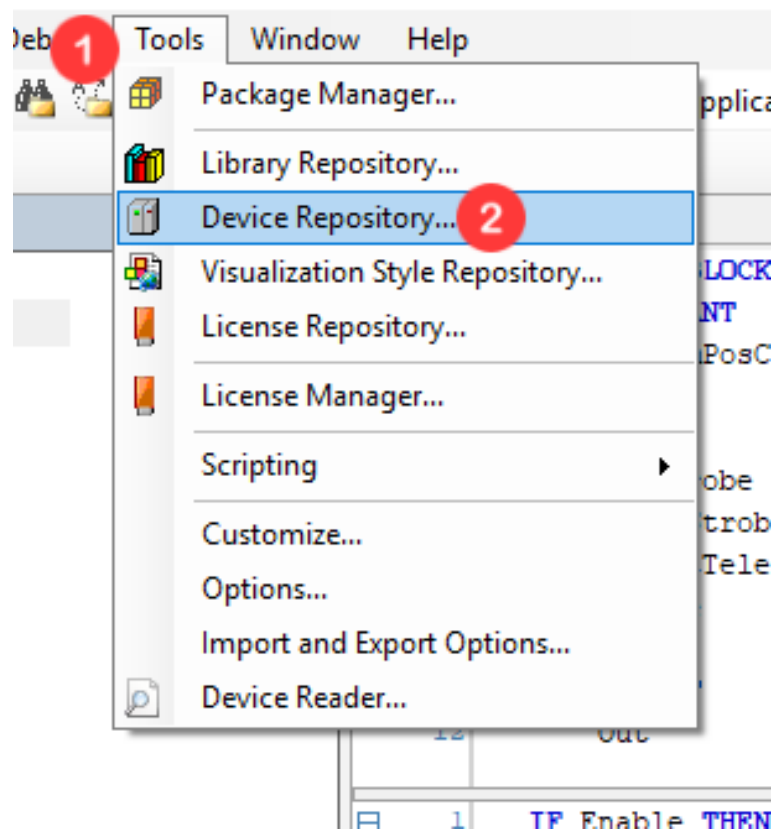
Import of a new device

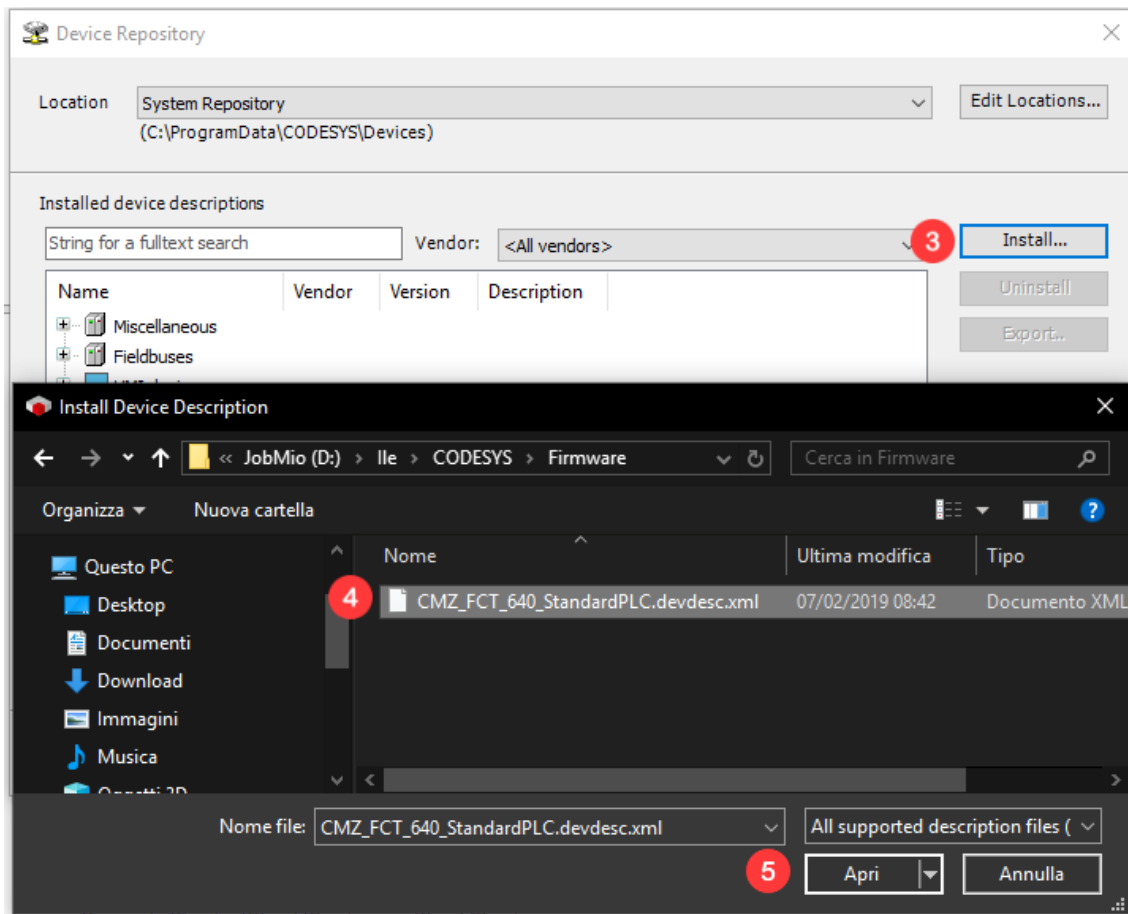
### Question

How shall the description file of a new device be imported in CODESYS?

### Answer

To import in CODESYS the description file in a new device follow the hereafter described steps:





- 1 From the menu bar click on *Tools*.
- 2 Select *Device Repository*.
- 3 Click on *Install...*
- 4 Select the description file of the device that has to be imported in CODESYS.
- 5 Click on *Apri*.

If the device description file to be imported is not at disposal, refer to the question [Descriptor file of the FCT controller](#) for an explanation on where to find it.

# Management of the CAN and ETC nodes startup

Gestione dello startup e reset dei nodi CAN e ETC

## Question

How shall the CAN and ETC nodes startup be managed in CODESYS?, through the library CMZ\_NodeUtilities?

## Answer

To correctly manage the nodes startup follow the hereafter described procedure:

The image shows two parts of the CODESYS interface. On the left, a project tree shows the configuration for a device (FCT200 Fieldbus Controller: SoftMotion). The 'CANopen Manager' settings are visible, with the 'Start Slaves' checkbox unchecked and 'NMT error behaviour' set to 'Stop Slave'. A red circle highlights the 'Start Slaves' checkbox. On the right, the 'General' settings for the CANopen Manager are shown, including Node ID (127), COB ID (Hex 16# 80), Cycle period (4000 us), and Window length (1200 us). Below this, a ladder logic program (PLC\_PRG) is shown. The program starts with a 'PROGRAM PLC\_PRG' block, followed by a 'VAR' block defining 'StartAxisCAN' as 'CMZ\_NodeUtilities.StartAxis' and 'ResetAxes' as 'BOOL'. The main logic block is 'StartAxisCAN(Axis := Axis, Enable := TRUE, ResetErrorStop := ResetAxes);', with comments indicating 'Axis reference' and 'Flag to reset error'.

- 1 From the *CANopen\_Manager\_SoftMotion* settings uncheck the *Start Slave* box and set *Stop Slave* in the *NMT error behaviour* option.
- 2 Use, as in the example, the function block *StartAxis* to manage the node start and reset.

## Link of a variable over a PDO

Link of a variable over a PDO

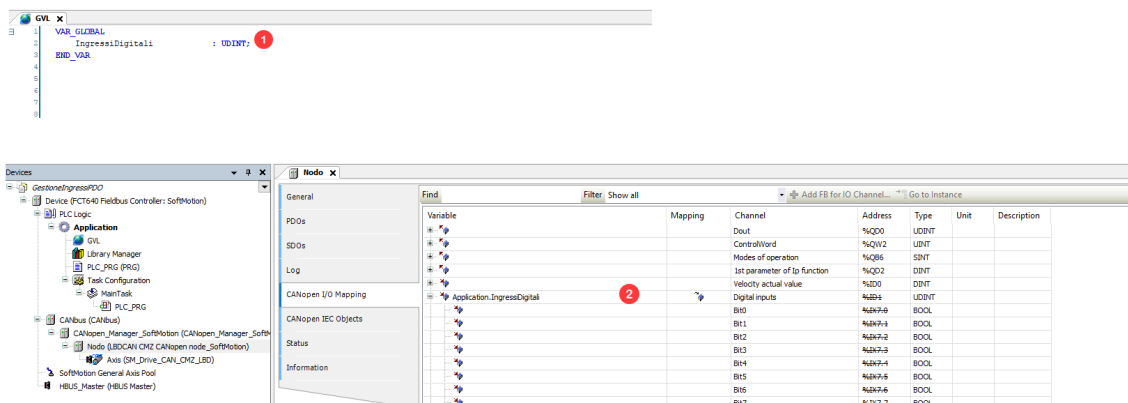
### Question

How shall a variable to be linked over a PDO read its value ?

### Answer

To read the entire PDO there are two link modes:

- Link of an already existing variable:



- 1 Create a variable that corresponds to the PDO type.
- 2 Enter the tab *CANopen I/O Mapping* between the node settings and link the created variable on the PDO, by selecting the variable by clicking on the icon (...) that appears when clicking on the column *Variable* in the PDO line where the variable has to be linked.

- Creazione e link di una variabile non esistente:

The screenshot shows the 'CANopen I/O Mapping' configuration window. The 'PDOs' table is visible, with a red circle highlighting the 'Variable' column. The table contains the following data:

| Variable       | Mapping | Channel                      | Address | Type  | Current Value | Prepared Value | Unit | Description |
|----------------|---------|------------------------------|---------|-------|---------------|----------------|------|-------------|
|                |         | Dout                         | %Q00    | UDINT | 0             |                |      |             |
|                |         | Controlword                  | %QW2    | UINT  | 0             |                |      |             |
|                |         | Modes of operation           | %Q96    | SINT  | 0             |                |      |             |
|                |         | 1st parameter of 1p function | %Q02    | DINT  | 0             |                |      |             |
|                |         | Velocity actual value        | %Q00    | DINT  | 0             |                |      |             |
| IngressDigital |         | Digital inputs               | %DI1    | UDINT | 0             |                |      |             |
|                |         | Statusword                   | %ZIV4   | UINT  | 0             |                |      |             |
|                |         | Modes of operation display   | %B10    | SINT  | 0             |                |      |             |
|                |         | Position actual value        | %Q03    | DINT  | 0             |                |      |             |

At the bottom of the window, the 'Watch 2' window shows the following data:

| Expression     | Application        | Type  | Value | Prepared value | Execution point   | Address | Comment |
|----------------|--------------------|-------|-------|----------------|-------------------|---------|---------|
| IngressDigital | Device-Application | UDINT | 0     |                | Cyclic Monitoring | %DI1    | Node 1  |

- Enter the tab *CANopen I/O Mapping* between the node settings and, by clicking on the column *Variable* in the PDO line, write the name of the variable that has to be created and linked. It will be created a variable of the same type of the PDO and at the same address, that can be seen by displaying the variable in the watch window. The variable can then be used in the program as it was a normal global variable created by the programmer.

the second link type must be used when every single bit of the PDO has to be read. It will be created a BIT type variable that can be used in the program as it was a normal global variable created by the programmer.

## Management of the strings in modbus

Management of the strings in modbus

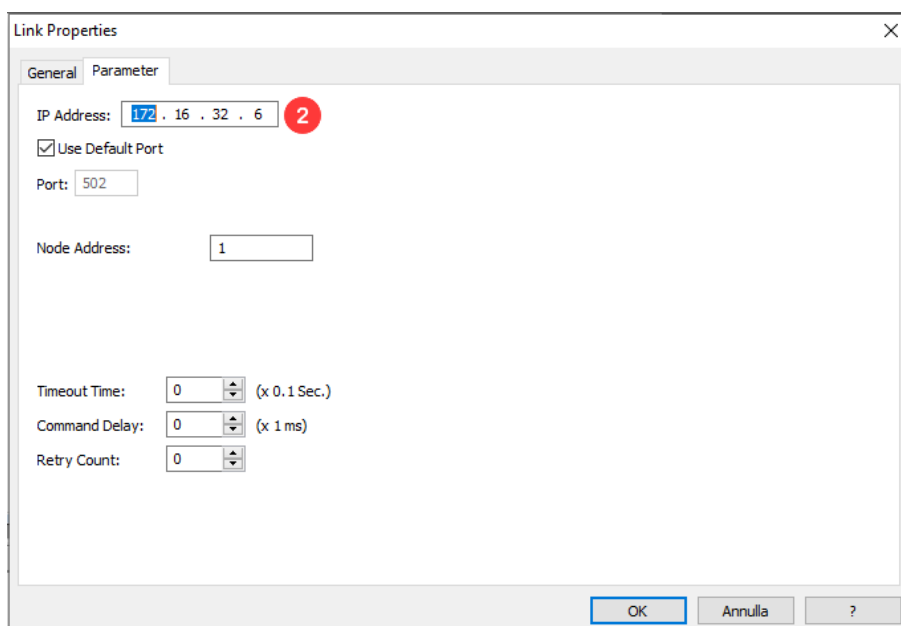
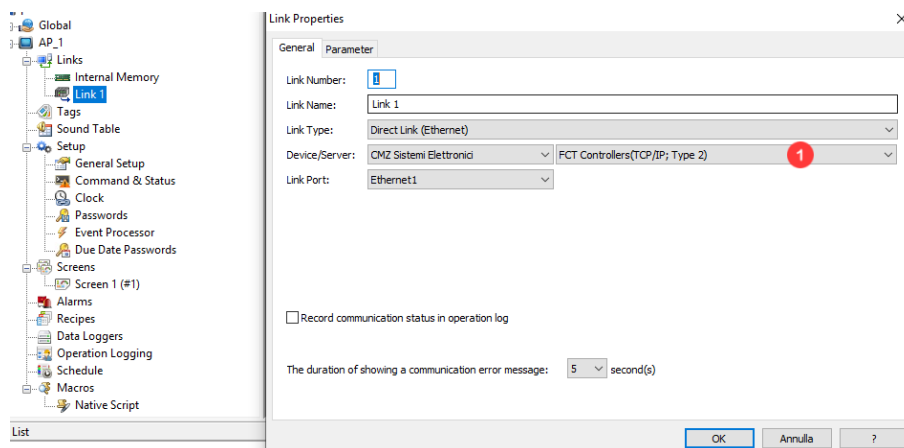
### Question

How shall the strings be managed in CODESYS-HMI through modbus?

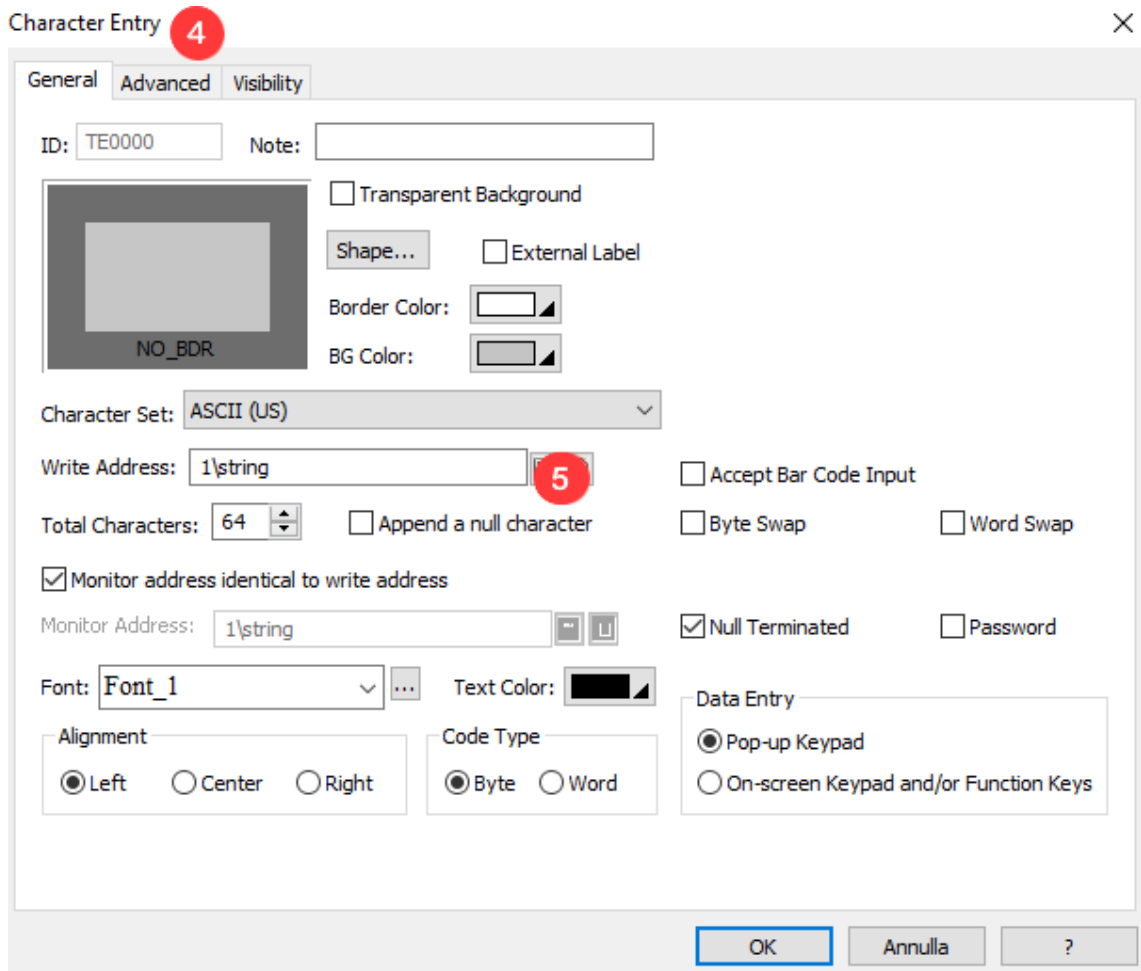
### Answer

In order to manage the strings the procedure is:

- From PM Designer:



|             |   |                       |         |        |
|-------------|---|-----------------------|---------|--------|
| provaString | 3 | 32-Bit Signed Integer | MB0_300 | Normal |
|-------------|---|-----------------------|---------|--------|



- 1 From the tree menu double click on *Link1* and from the window *Link properties* that pops up, in the tab *General*, select the connection type and the device type to which connect.
  - 2 From the window *Link properties*, in the tab *Parameter*, insert the address of the device to connect with.
  - 3 Create a tag with MB address.
  - 4 Import in the screen an object of type *Character Entry*.
  - 5 Select as *Write Address* the previously created tag and set the number of the characters that can be written and visualized.
- From CODESYS:

```
PROGRAM PRG_Modbus
VAR
  ModbusSlaveTcp_Inst : CMZ_Modbus.SlaveTcp();
  Flag : BOOL := TRUE;
END_VAR

IF Flag THEN
  Flag := FALSE;
  ModbusSlaveTcp_Inst.Map(1,CMZ_Modbus.DataTypeMap.HoldingRegistersDWord, 100,99, ADR(ArDWord), FALSE);
  ModbusSlaveTcp_Inst.Start();

END_IF
ModbusSlaveTcp_Inst.Work();
```

```
provaString AT %MB300 : STRING; 2
```

- 1 Import in the project the library *CMZ\_Modbus* and map the modbus array.
- 2 In the global variables declare a STRING type variable at the same address (%MB) of the tag that has been previously created on PM Designer.



# Activation of the FTP server with library CMZ\_FTPServer

Activation of the FTP server with library CMZ\_FTPServer

## Question

How shall the FTP server be activated by using the library CMZ\_FtpServer?

## Answer

After the library *CMZ\_FtpServer* has been imported in the project it is necessary:

```
PRG_FTP x
1 PROGRAM PRG_FTP
2 VAR
3   FtpRef      : CMZ_FtpServer.FTPSRV; // Reference del Server FTP
4   FtpSettingsCfg : CMZ_FtpServer.SETTINGS; 1 // Instance the structure to Server FTP configuration
5   Ftp_Create   : CMZ_FtpServer.Create;
6   Ftp_Init     : CMZ_FtpServer.Init;
7   Ftp_Go      : CMZ_FtpServer.Go;
8   Init        : BOOL;
9 END_VAR

1 // Gestione FTP
2 FtpRef.Instance := 0; 2 //Server FTP instance
3 FtpSettingsCfg.Port_Number := 0; //Set the default user configuration
4
5 //Creation of FTP server
6 Ftp_Create(FtpSrvRef := FtpRef, Execute := TRUE); 3
7 //Init of FTP server
8 Ftp_Init(FtpSrvRef := FtpRef, Execute := Ftp_Create.Done, Set := FtpSettingsCfg); 4
9 //Start of FTP server
10 Ftp_Go(FtpSrvRef := FtpRef, Execute := Ftp_Init.Done, Done => Init); 5
11
```

- 1 To define the following structures and function blocks:
- 2 Instance the FTP server and prearrange the default configuration of the users accounts.
- 3 Use the *Create* function to create the FTP server.
- 4 Use the *Init* function to initialize the FTP server.
- 5 Use the *Go* function to warn the FTP server.

## User creation for the access through FTP server

User creation for the access through FTP server

### Question

How shall an user be created to access to a particular folder through FTP server?

### Answer

After the library *CMZ\_FtpServer* has been imported in the project, and that the FTP server has been created, initialized and executed through the function blocks *Create*, *Init* and *Go*, it is necessary to follow the hereafter described steps to create a new user:

```
DoneOp           : DINT;  
FTPUser          : CMZ_FtpServer.USER_SETTINGS;  
NomeUtente      : STRING := 'prova';  
PasswordUtente  : STRING := 'prova';  
HomePathUtente  : STRING := 'A:/CODESYS/';  
ProprietaUtente : WORD := CMZ_FtpServer.Constants.USER_SETTINGS_PROPERTY_DEFAULT;
```

```
FTPUser.User_Name := NomeUtente;  
FTPUser.Password := PasswordUtente;  
FTPUser.Home_Path := HomePathUtente;  
FTPUser.Properties := ProprietaUtente;
```

```
DoneOp := CMZ_FtpServer.AddUser(FTP, FTPUser);
```

- 1 Define the following variables and structures.
- 2 Use the previously defined structure and characterize the user by selecting the user name, the password, the path to access and the properties.
- 3 Use the *AddUser* function, that is inside the library, to create the user.

# Axis resolution

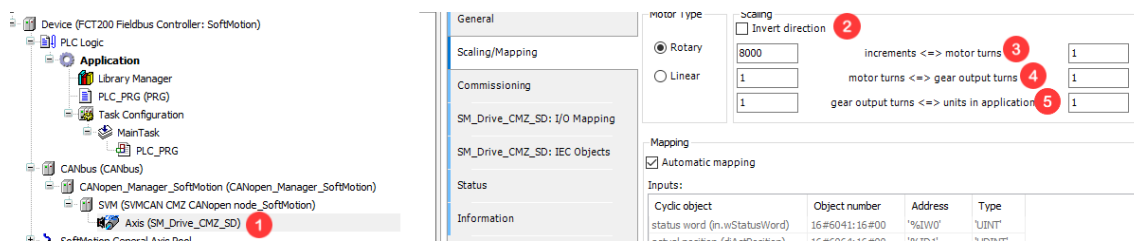
Axis resolution

## Question

How shall the axis resolution be set from the program?

## Answer

To set the axis resolution follow the hereafter described steps:



- 1 Double click on the axis on which the resolution has to be set and enter the tab *Scaling/Mapping*.
- 2 Set the resolutions that are needed for the axis.
- 3 Number of increments that corresponds to a defined motor revolutions.
- 4 Number of motor revolutions that corresponds to a defined revolutions number on the motor shaft output.
- 5 Number of the revolutions number on the motor shaft output that corresponds to a defined unit in the application.

# Connection to the FCT without network scan

Connection to the FCT without network scan

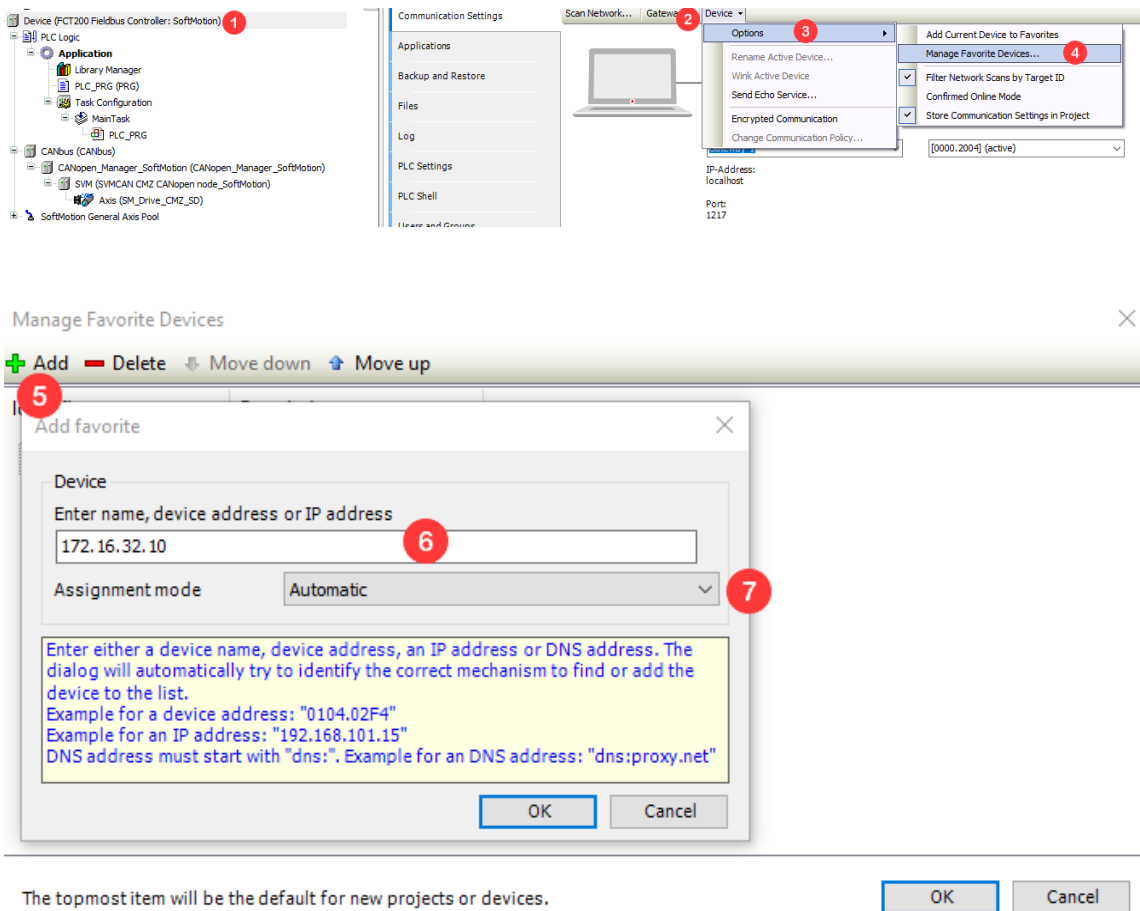
## Question

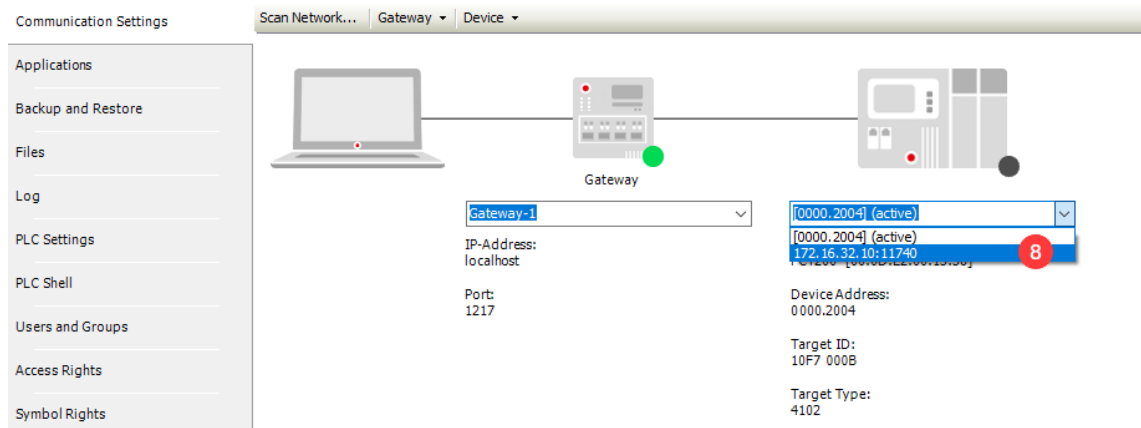
Which are the steps to connect to the FCT from CODESYS with network scan?

## Answer

The modes to connect to the FCT without to scan the network are:

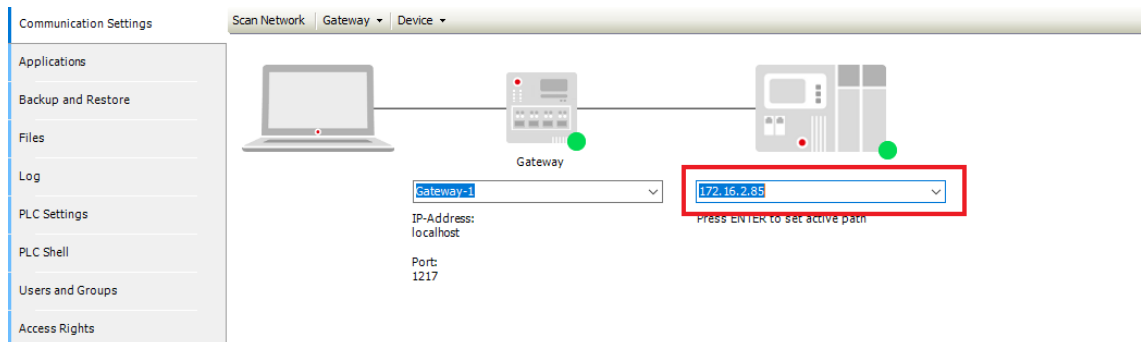
- Add a device with a determined IP address by following the hereafter described steps:





- 1 Double click on the device and enter the tab *Communication Settings*.
- 2 Click on *Device*.
- 3 Click on *Option*.
- 4 Click on *Manage Favorite Devices...*
- 5 From the window *Manage Favorite Devices...* click on *Add*.
- 6 Insert the address of the FCT to which connect.
- 7 As *Assignment mode* select *Automatic*.
- 8 From the tab *Communication Settings* select the FCT with which connect, previously added.

- Without adding a device directly insert the controller IP address, as follows, and push *Enter*:



## Management of the modbus TCP (client FCT) with library CMZ\_Modbus

Management of the modbus TCP (client FCT) with library CMZ\_Modbus

### Question

How shall the communication modbus TCP with client FCT and the data sending and receiving be started?

### Answer

To manage the communication modbus TCP, having the FCT as client, it is necessary to import the library *CMZ\_Modbus* in the project and:

```

PROGRAM PRG_ModbusMaster
VAR
    ModbusMasterTcp Inst : CMZ_Modbus.MasterTcpPort; 1
    Flag                : BOOL := TRUE;
END_VAR



---


IF Flag THEN
2    Flag := FALSE;
    ModbusMasterTcp_Inst.szIPAddress := '192.168.0.10';
    ModbusMasterTcp_Inst.uiPort := 502; 3
END_IF

ModbusMasterTcp_Inst();

```

- 1 to instance the function block *MasterTcpPort* that manages the TCP port.
- 2 To use the method *szIPAddress* to set the address of the server to which connect.
- 3 To use the method *uiPort* to set the number of the port to be opened.

Use the following function blocks that are provided by the library to read and write the data according to the data type:

- MasterReadCoils
- MasterReadDiscreteInputs

- MasterReadHoldingRegisters
- MasterReadInputRegisters
- MasterReadWriteRegisters
- MasterWriteMultipleCoils
- MasterWriteMultipleRegisters
- MasterWriteSingleCoils
- MasterWriteSingleRegister

```
    ReadRegister      : CMZ_Modbus.MasterReadHoldingRegisters;  
    WriteRegister    : CMZ_Modbus.MasterWriteSingleRegister;  
END_VAR
```

---

```
ReadRegister(pMbPort := ADR(ModbusMasterTcp),  
            byNodeId := 1,  
            pData := ADR(pData),  
            wAddress := StartAddressToRead,  
            byCount := NumRegisterToRead,  
            pData := DataRead  
            );  
  
Writeregister(pMbPort := ADR(ModbusMasterTcp),  
            byNodeId := 1,  
            wAddress := 1,  
            wValue := NewTagID,  
            wAddress := StartAddressToWrite,  
            wValue := ValueToWrite  
            );
```

Figure 8.5. Example of use of the function blocks to read and write holding registers

## Management of the modbus TCP (server FCT) with library CMZ\_Modbus

Management of the modbus TCP (server FCT) with library CMZ\_Modbus

### Question

How shall the modbus arrays be mapped in CODESYS, by using the CMZ library?

### Answer

To map the modbus arrays it is necessary to import the library *CMZ\_Modbus* in the project and follow the example below:

```

C_PRG | PRG_Modbus x
PROGRAM PRG_Modbus
VAR
  ModbusSlaveTcp_Inst : CMZ_Modbus.SlaveTcp();
  Flag : BOOL := TRUE;
END_VAR

IF Flag THEN
  Flag := FALSE;
  ModbusSlaveTcp_Inst.Map(1, CMZ_Modbus.DataTypeMap.HoldingRegistersWord, 0, 99, ADR(ArWord), FALSE); 1
  ModbusSlaveTcp_Inst.Map(1, CMZ_Modbus.DataTypeMap.HoldingRegistersDWord, 100, 99, ADR(ArDWord), FALSE); 2
  ModbusSlaveTcp_Inst.Start();
END_IF
ModbusSlaveTcp_Inst.Work();

```

1 Call the holding register instance of Word type:

- *CMZ\_Modbus.DataTypeMap.HoldingRegistersWord*: data type (holding registers word)  
0: starting modbus address (always expressed in word).
- 100: Number of mapped elements of word type.
- *ADR(ArWord)*: array address, declared in the GVL, in which the modbus variables have to be mapped.
- *FALSE*: Swap byte register.

2 Call the holding register instance of DWord type:

- *CMZ\_Modbus.DataTypeMap.HoldingRegistersDWord*: data type (holding registers dword).



*0* : starting modbus address (always expressed in word).

- *100* : Number of mapped elements of dword type.
- *ADR(ArDWord)* : array address, declared in the GVL, in which the modbus variables have to be mapped.
- *FALSE* : Swap word register.

# Activation of the analog inputs of the WAGO module

Activation of the analog inputs of the WAGO module

## Question

How shall the analog inputs of the WAGO module be activated from CODESYS?

## Answer

To activate the analog inputs of the WAGO module, follow the settings as showed in the image below: *Figure 8.6*.

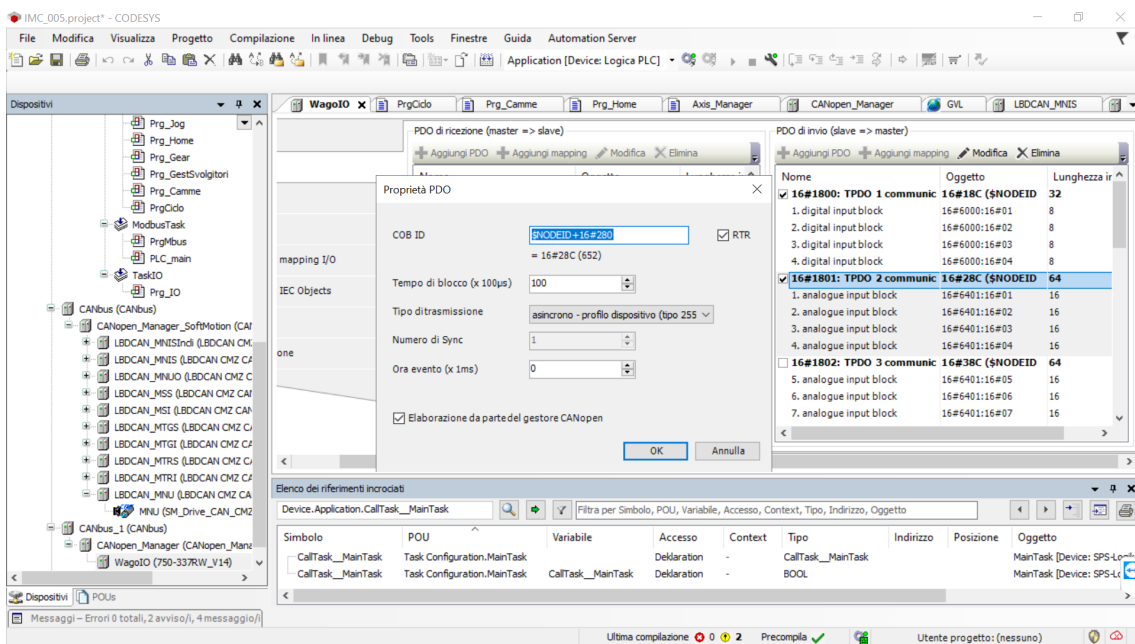


Figure 8.6. Analog inputs settings

## Management of retentive modbus variables

Management of retentive modbus variables

### Question

How shall the retentive modbus variables be managed from CODESYS?

### Answer

To manage the modbus variables from CODESYS, it is necessary to follow the hereafter reported steps:

```

GVL_Modbus x
1  {attribute 'qualified_only'}
2  VAR_GLOBAL
3      Ar1 AT %MB0           : ARRAY [0..99] OF WORD;
4      Ar2 AT %MB200        : ARRAY [0..99] OF DWORD;
5
6      varModbus AT %MB200   : DINT; 1
7  END_VAR

```

```

GVL_Retain x
1  {attribute 'qualified_only'}
2  VAR_GLOBAL RETAIN
3      varRetain            : DINT; 2
4  END_VAR

```

```

IF Flag THEN
  Flag := FALSE;
  ModbusSlaveTcp_Inst.Map(1,CMZ_Modbus.DataTypeMap.HoldingRegistersWord, 0,100, ADR(GVL_ModBus.Ar1), FALSE);
  ModbusSlaveTcp_Inst.Map(1,CMZ_Modbus.DataTypeMap.HoldingRegistersDWord, 100,200, ADR(GVL_ModBus.Ar2), FALSE);
  ModbusSlaveTcp_Inst.Start();

  GVL_Modbus.varModbus := GVL_Retain.varRetain; 3
END_IF
ModbusSlaveTcp_Inst.Work();

IF ModbusSlaveTcp_Inst.xStarted AND ModbusSlaveTcp_Inst.Error = CMZ_Modbus.ModbusErrors.noerror THEN
  GVL_Retain.varRetain := GVL_Modbus.varModbus; 4
END_IF

```

- 1 Declare the modbus variable.
- 2 Declare the corresponding retentive variable.
- 3 Before to start the modbus, copy the retentive variable value in the corresponding modbus variable.
- 4 When the modbus is correctly started, copy the value of the modbus variable in the corresponding retentive variable.

## Comparison between two arrays

---

Comparison between two arrays

### Question

How shall two arrays be compared in CODESYS?

### Answer

To compare two arrays in CODESYS it is provided the *Compare* function that is present in the library *CAA Memory*. This function requires as input the two pointers to the arrays that it has to compare and the number of bytes to be compared, and returns the position of the first value that it finds different between the two arrays.

```
abyMemoryBlockA : ARRAY[0..9] OF BYTE := 0,1,2,3,4,5,6,7,8,9 ;  
abyMemoryBlockB : ARRAY[0..9] OF BYTE := 0,1,2,0,4,5,6,7,8,9 ;  
  
MEM.Compare(ADR(abyMemoryBlockA),ADR(abyMemoryBlockB),10) = 4
```

Figure 8.7. Example of use

## Set, not automatically, the ID of an EtherCAT node

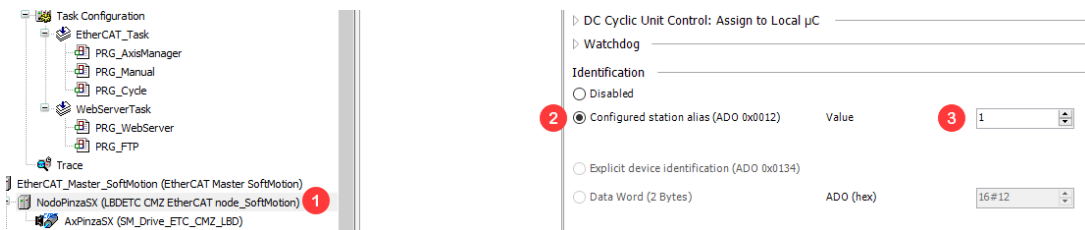
Set, not automatically, the ID of an EtherCAT node

### Question

How shall the ID of an EtherCAT node be set, not automatically, in a project?

### Answer

To set the ID of an EtherCAT node follow the hereafter described steps:



- 1 Double click on the node on which the node ID has to be changed, to enter the node settings.
- 2 Set, between the node settings, the *Configurated station alias* option.
- 3 Write the node ID that shall be seto to the node.

## Activation of the web server and api management with library CMZ\_WebServer

---

Activation of the web server and api management with library CMZ\_WebServer

### Question

How shall the web server be activated and how shall the api that contains the variables to be sent/received be managed?

### Answer

To activate the web server and to manage the api it is necessary to:

```
PROGRAM PRG_WebServer
VAR
    FB_WebServer      : CMZ_WebServer.WebServer;
    FB_WebServer(Enable := TRUE,
                 WebRoot := 'WebServer\'',
                 pFB_UserAPIExe := ADR(MyFB_UserApiExe)
    );
```

```

MyFB_UserApiExe x
1  FUNCTION_BLOCK MyFB_UserApiExe EXTENDS CMZ_WebServer.UserAPIExe 2
2  VAR_INPUT
3  END_VAR
4  VAR_OUTPUT
5  END_VAR
6  VAR
7  END_VAR
8
9
10
11
12
13
14
15  IF API_Name = 'Gruppo1' THEN
16
17      pManageBufferTCP^.OpenObj();
18
19      pManageBufferTCP^.SendRecv_INT('VAR1', ADR(VAR1));
20      pManageBufferTCP^.SendRecv_INT('VAR2', ADR(VAR2));
21
22      pManageBufferTCP^.CloseObj();
23      Done := TRUE;
24
25  ELSIF API_Name = 'Gruppo2' THEN
26
27      pManageBufferTCP^.OpenObj();
28
29      pManageBufferTCP^.SendRecv_BOOL('NetOk', ADR(NetOk));
30      pManageBufferTCP^.SendRecv_BOOL('StatoErrorAxTotal', ADR(StatoErrorAxTotal));
31      pManageBufferTCP^.SendRecv_BOOL('StatoEnabledAxTotal', ADR(StatoEnabledAxTotal));
32
33      pManageBufferTCP^.CloseObj();
34      Done := TRUE;
35
36  ELSIF API_Name = 'Array' THEN
37
38      pManageBufferTCP^.OpenObj();
39
40      FOR iArr := 0 TO 9 DO
41          varName :=CONCAT('Array[' , TO_STRING(iArr));
42          varName :=CONCAT(varName, ']');
43          pManageBufferTCP^.SendRecv_DINT(varName, ADR(ArrayVal[iArr]));
44      END_FOR
45
46      pManageBufferTCP^.CloseObj();
47
48      Done := TRUE;
49  END_IF;

```

1 instance the *WebServer* function block after having imported the *CMZ\_WebServer* library, by managing:

- the *Enable* input to enable the web server.
- The *WebRoot* input to indicate where is, in the controller, the folder for the web server (with the eventual html pages...).
- The *pFB\_UserAPIExe* input indicating the pointer to the function block that contains the api.



- 2 Create a function block that extends the *UserAPIExe* function block.
- 3 Inside this function block create the necessary api by managing, through the *SendRecive\_tipodato* method (SendRecive\_BOOL, SendRecive\_INT...), the variables to be transferred or received.

## H\_Bus starting problems

---

H\_Bus starting problems

### Question

Why the H\_Bus does not switch to operational?

### Answer

A reason why the H\_Bus does not switch to operational is that in the CODESYS project the modules are not correct or have not been inserted in the correct order.

| Code     | Description                           |
|----------|---------------------------------------|
| HMI_0001 | Retentive variables on HMI            |
| HMI_0002 | Communication between HMI and SDDrive |
| HMI_0003 | Modify the keyboard dimension         |
| HMI_0004 | Trasfer of a project between two HMI  |

**Table 9.1. Arguments**

## Retentive variables on HMI

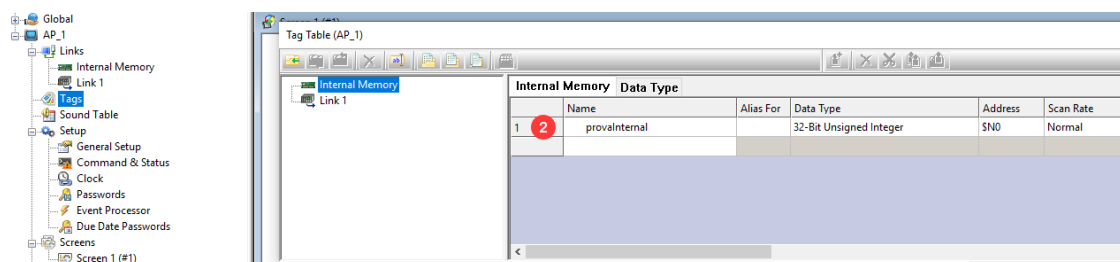
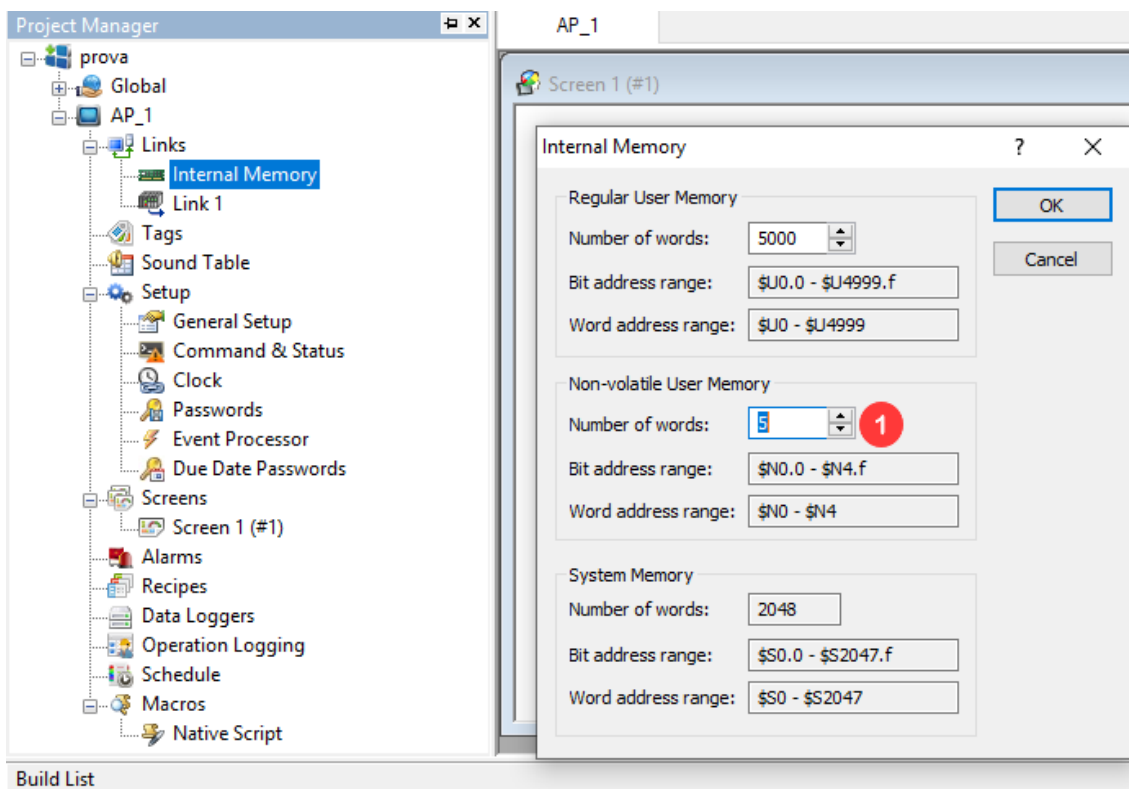
Retentive variables on HMI

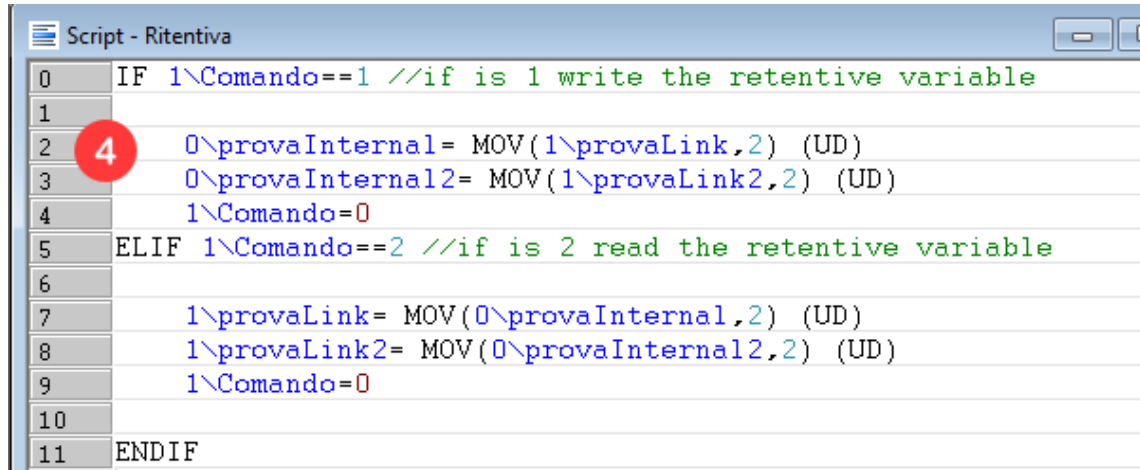
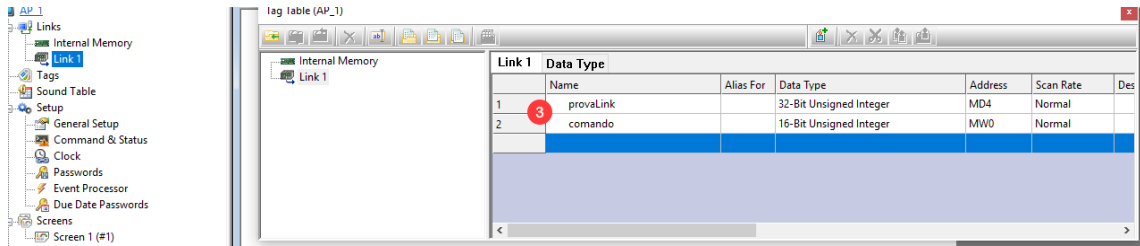
### Question

How shall the variables in the Cermate panels be transformed in retentive in the PM Designer?

### Answer

To transform the variables in retentive through the Cermate panel follow the hereafter described steps:





- 1 From the tree menu double click on *Internal Memory* and define, through the *Number of words* cell in the *Non-volatile User Memory* section, the number of the retentive words that has to be used
- 2 From the tree menu double click on *Tags* and in the *Internal Memory* section define as much tags as much are the variables that have to be transformed in retentive in the program.  
It is necessary to use the addresses  $\$N$ .
- 3 From the tree menu double click on *Tags* and in the *Link* section define:
  - As many tags as are the variables that have to be transformed in retentive in the program. These tags will act as a go-between the internal memory defined tags and the program variables.
  - A tag that allow to read or write the program variables, according to the value that has been passed by the program.
- 4 Create a macro that writes or reads the retentive variables, according to the value of the previously defined tag *Comando*.

Through the *MOV* function the value of a variable stored in a memory area is copied in another memory area, defined by the user.

The user will manage from the program, according to what he needs, the variable that is linked to the tag *Comando*. It will be even possible to save the variables in the non-volatile memory areas or, vice versa, it will be possible to load in the program variables the corresponding value, previously saved in the non-volatile memory areas.

According to the example, when it is assigned the value 1 to the variable *Comando* from the program, the actual value of the variable is saved in retentive mode, while if it is assigned the value 2 the value of the retentive variable is loaded in the program variable (this operation shall be done at every turn on of the device).

# Communication between HMI and SDDrive

Communication between HMI and SDDrive

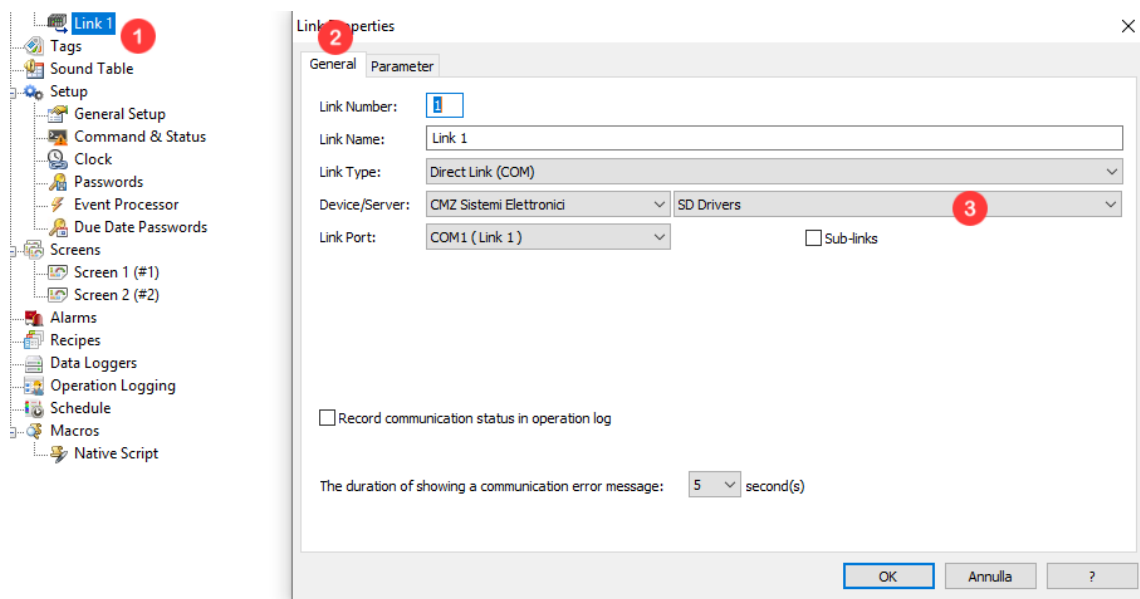
## Question

Why do the panel and the SDDrive not correctly communicate?

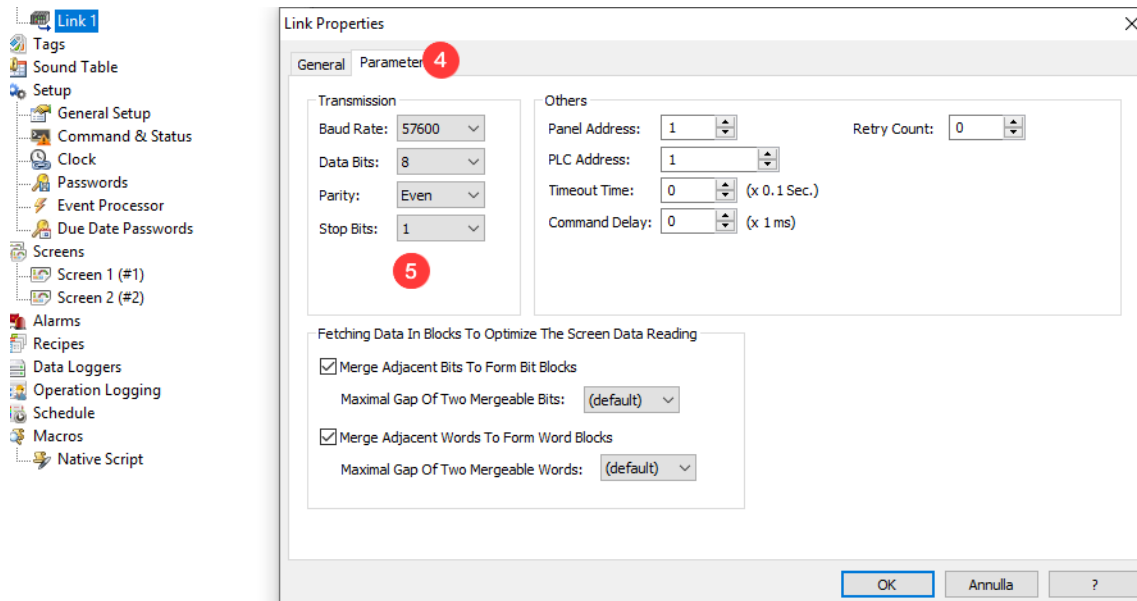
## Answer

After the check of the correct connection between HMI and SDDrive has been done, it is necessary to verify that the communication characteristics are correct both on the HMI side and on the drive side.

On HMI side:

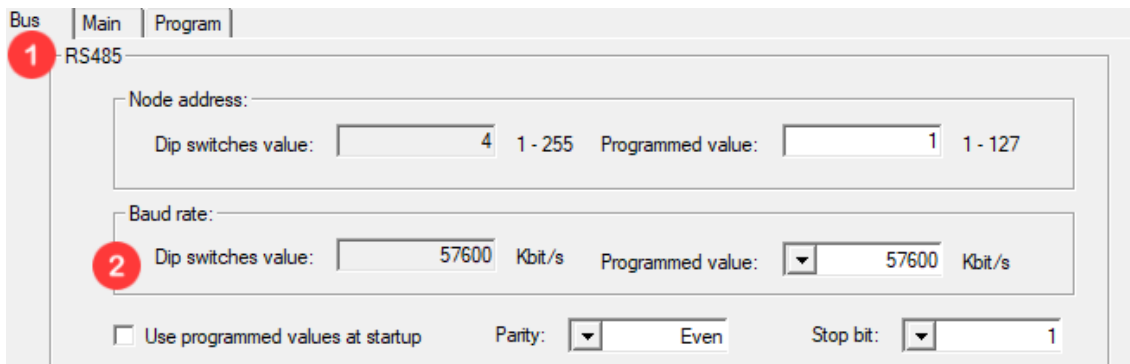


- 1 From the tree menu double click on *Link1*.
- 2 From the window that contains the link properties enter the tab *General*.
- 3 Set the communication settings as in the image, to communicate with the SDDrives.



- 4 Enter the tab *Parameter*.
- 5 Check that the communication values correspond to the ones set in the drive.

From SDSetup:



- 1 From SDSetup enter the tab *Bus*.
- 2 Check that the communication values correspond to the ones set on PM Designer.



# Modify the keyboard dimension

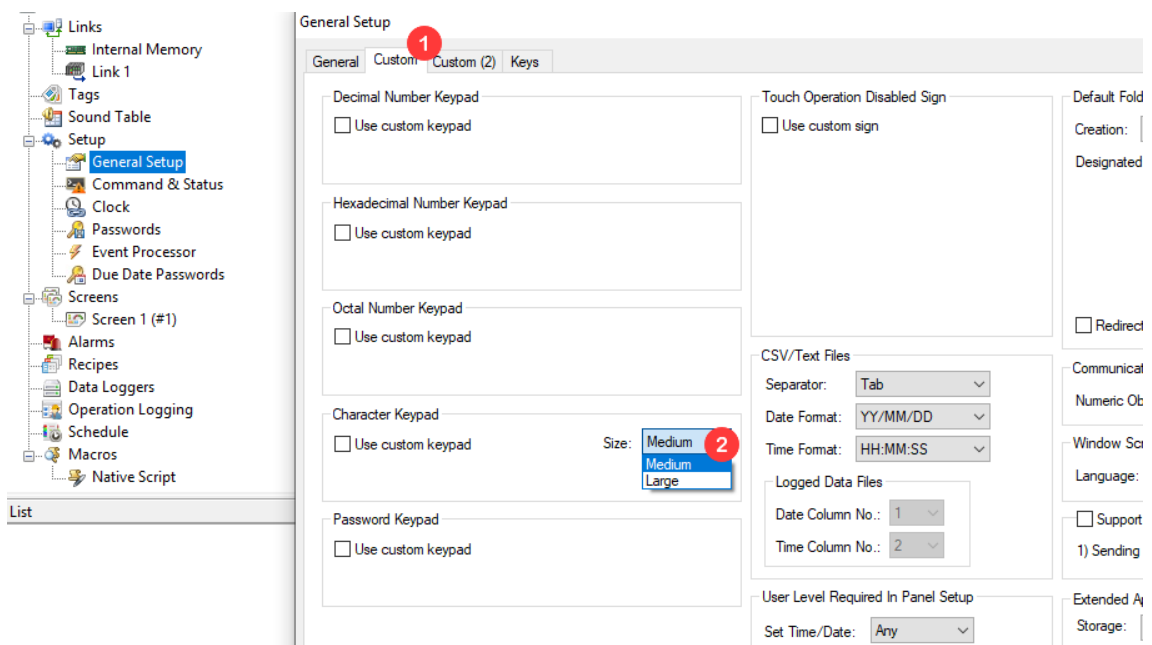
Modify the keyboard dimension

## Question

How shall the keyboard dimension be modified through PM Designer?

## Answer

To modify the dimension of the keyboard follow the following steps:



- 1 From the tree menu double click on *General Setup* and enter the tab *Custom*.
- 2 From the *Size* option select the desired dimension between *Medium* and *Large*.

## Trasfer of a project between two HMI

---

Trasfer of a project between two HMI

### Question

How shall the transfer of a project be done between two HMI?

### Answer

To transfer a project from an HMI to another it is necessary to:

- if the model of the two HMI **is not the same**, open the project with PMDesigner and convert it with the version that is suitable for the recipient HMI.
- if the model **is the same** it is possible to transfer the project:
  - with an usb key: plug-in the key behind the HMI in the provided port, in the *Panel Setup* page (which is accessible if during the turn on the upper-right corner of the HMI is pressed) push the button *Copy to File* and select the folder in the key in which the file has to be saved.  
After the file has been saved, it is necessary to plug-in the key in the recipient HMI and, from the *Panel Setup* page, push the button *Update from File* and select the project to be downloaded in the HMI.
  - Directly between HMI and HMI: connect the two HMI and then, from the *Panel Setup* page, push the button *Copy to HMI*.





CMZ reserves the right to change the data in order to update or improve its products without prior notice  
CMZ si riserva il diritto di modificare i dati per aggiornare o migliorare i propri prodotti senza alcun preavviso

soga  energyteam

CMZ SISTEMI ELETTRONICI SRL

Via dell'Artigianato 21  
31050 Vascon di Carbonera (TV)  
Italy  
Phone +39 0422 447411  
Email [cmz@cmz.it](mailto:cmz@cmz.it)

[cmz.it](http://cmz.it)